修士学位論文

麻雀類似ゲームの学習アルゴリズムに関する研究

2020 年度

広域科学専攻　広域システム科学系

31-196812

清水大志

目次

第 1 章

はじめに

Texas hold'em [1] や StarCraft2 [2]，Dota2 [3] などいくつかの多人数不完全情報ゲームにおいて，コンピュータプレイヤの作成に強化学習を用いる手法が提案されている．その中には作成されたコンピュータプレイヤが人間の上級者の実力を超えたと主張する研究もある．例えば麻雀では，オンラインゲームの天鳳[*1]においてコンピュータプレイヤで初めて十段を達成した Super Phoenix [4]（以下 Suphx と表記する）があるが，学習の始めに人間の牌譜を使った教師あり学習を行なっている．

手生成の特徴量（hand-crafted features）や上級者の牌譜といったゲームに関する人間の事前知識を使わずに，麻雀において自己対戦による強化学習のみで人間の上級者を超える実力を得ることを本研究の目標とする．ゲームにおいて人間の事前知識を使わずに人間の上級者の実力を超えた例としては，上記の例の他に AlphaGo Zero [5] がある．畳み込みニューラルネットワークとモンテカルロ木探索を組み合わせて作られた元のプログラム AlphaGo [6] を大きく超える強さを，人間の棋譜を用いずに自己対戦による強化学習のみで AlphaGo Zero は達成した．この成功を受け，他のゲームでもゲーム固有の特徴量をなるべく入力に使わないニューラルネットによる方策（policy）関数や価値関数の学習が試みられている．麻雀に関しては，築地ら [7] や Gao ら [8] による先行研究があるが，どのような入力を与えるのが適切か，またどのようなネットワークモデルが適しているかについては，十分な検討がおこなわれていない．その原因として，先行研究の多くが人間の牌譜を用いた教師あり学習を使っているために，多くのエラー含む人間の牌譜を用いることによる教師データの質の確保が難しい点，学習に必要な局面数を増やすのが難しい点などが考えられる．局面数を増やしたり教師データの質を確保する解決策の一つは自己対戦による強化学習がある．しかし，強化学習は実行に時間がかかり，また，不適切なネットワークモデルではいつまで経っても学習が進まない．

ここで，適切なネットワークモデルについても，麻雀とその類似ゲームが共通する性質をもっているという仮定に基づいて，小さな麻雀類似ゲームについてネットワークモデルの評価を行った．小さいゲームに対する教師あり学習は短い時間で終了する．そのため Optuna [9] などのハイパー

[*1] `https://tenhou.net/`

パラメータ自動最適化ツールを用いることによって，幅広いネットワークモデルの中から通常は人の手で最適化されるハイパーパラメータを含めて適したモデルを選択することが可能となる．その例として本研究では，麻雀のゲームとしての特徴を保持しつつ，理論的な最善手が求められるミニゲーム（以下ではミニ麻雀 [10] と呼ぶ）を対象として教師あり学習によりネットワークモデルを評価した．また，ニューラルネットワークを用いたモデルにおいては手牌をどのように入力に変換するのかという点もモデルの学習速度に大きく影響する．麻雀の手牌を表現する方法は複数考えられるが，これも十分な検討が行われていない．そこで本研究では三種類の表現方法を考えて比較を行なった．

ミニ麻雀で得られたモデルや手牌の表現方法を踏まえて，ミニ麻雀よりも扱う牌の種類や役が増えてより通常の麻雀に近づいたすずめ雀 [11] を次に扱い，これに強化学習を適用した [12]．すずめ雀は二人から五人で行う麻雀類似ゲームで，「麻雀で使う牌の種類や手牌の枚数が減っている」「鳴きが存在しない」といった点で，麻雀のルールを単純化したゲームである．このゲームを対象にすることによって，学習が短い時間で進み，適用手法の効果の検証が行いやすくなるという効果が期待される．すずめ雀では役が多くあり，また実際の麻雀に近くなるようゲームバランスも考慮されているため，ミニ麻雀より実験環境に適していると判断した．この実験では，手生成の特徴量（hand-crafted features）や上級者の牌譜といったゲームに関する人間の事前知識をなるべく使わずに強いすずめ雀プレイヤを作成することを目標とする．はじめに，自分の手牌のみを考慮に入れて割引収益の期待値が最も高い牌を切ることができる一人すずめ雀プレイヤを作成した．次に，このプレイヤを対戦相手として強化学習を行い，一人すずめ雀プレイヤに迫る強さのプレイヤを作成できた．最後に，各局の点数の最大化を目指すのではく，全局を終えたときの平均順位の更なる改善を目的として，Global Reward Prediction による予測値を報酬に用いる試みを行った．これは Super Phoenix で提案された強化学習により良い報酬関数を作成する手法であるが，この手法による平均順位の改善は達成できていない．これには報酬関数や一人すずめ雀プレイヤの強さが影響していると考えられる．

以下，第 2 章では麻雀の基本的なルールとその用語について，第 3 章では強化学習を用いた麻雀プレイヤの作成に関する主要な関連研究について，第 4 章ではミニ麻雀のルールとそれを用いた教師あり学習の方策のモデル評価の実験について，第 5 章ではすずめ雀のルールとそれを対象とした実験について，第 6 章ではまとめと今後の課題について述べる．

第 2 章

麻雀について

この章においては，麻雀の基本的なルールや重要な用語について簡単に説明する．麻雀のルールは非常に複雑であり，それを全て理解することは本研究の理解には必要ないことであるため，ルールについては最小限の説明にとどめ，今後出てくる主な用語の説明を行う．なお，麻雀は国によって使用する牌の枚数も含めてルールが違うが，以下では日本で一般的に遊ばれているルールについて述べる．

麻雀は主に四人で行われる不完全情報ゲームの一種である．ゲームは複数の**局**からなり，局の終了時にはその局の結果に基づいてプレイヤ同士の得点の授受が行われる．最終局が終了した時に各プレイヤの得点に基づいて順位を決定する．

麻雀では図 2.1 に示す**萬子**（マンズ），**筒子**（ピンズ），**索子**（ソーズ），**字牌**と呼ばれる四種類の麻雀牌を用いる．萬子と筒子，索子はそれぞれ 1 から 9 までの九種類の牌が，字牌は「東」「南」「西」「北」「白」「發」「中」という七種類の牌がある．索子や萬子，筒子のような数字を表す牌を**数牌**という．同一牌がそれぞれ四枚ずつ入っており，全部で 136 枚の牌を用いるのが一般的なルールである．各プレイヤは 13 枚または 14 枚の麻雀牌を**手牌**として局の始めに引き，残りの牌を**壁牌**（ピーパイ，**山**とも）として残す．山から一枚牌を引き，不要な牌を一枚捨てるという行為を各プレイヤは繰り返し順番に行い，局が終了する条件を満たすまで続ける．局が終了する条件として一般的なものは，誰かが**和了する**（ホーラする，**あがる**とも）ことと，山がなくなることである．なお，山は全て引ききるのではなく，14 枚は残しておく．

あがるとは特定の**役**が少なくとも一つ（ある特定の条件よっては二つ以上）がある状態で**あがりの形**を構成することである．プレイヤが手牌を特定の牌のみで構成するといった手牌構成の条件や，ある特定の状況に自動的に付与される偶然的な条件のことを役という．また，あがりの形とは基本的に四つの**面子**と一つの**雀頭**を構成することである．面子は**順子**（シュンツ）と**刻子**（コーツ），**槓子**（カンツ）の三種類がある．順子は「123」のように同じ種類の牌の三つの連番のこと，刻子は「111」のように同一牌三つのこと，槓子は「3333」のように同一牌四つのことを指す．雀頭は「22」のように同一牌二つのことを指す．

プレイヤは他者の捨て牌を一枚もらって面子を構成することができ，これを**鳴き**という．ポンは刻子を構成する時に，チーは順子を構成する時に，カンは槓子を構成する時に行う．あがりには**ロ**

ンあがりとツモあがりがある．ツモあがりはあがりに必要な最後の牌を山から引くことであり，ロンあがりは他プレイヤの捨て牌からもらうことである．全プレイヤのうち一人は親，残りのプレイヤは子となる．親のあがり時に得られる得点は子の約 1.5 倍である．子がツモあがりした時に親が支払う点数は，他の子の約 2 倍である．

その他に本研究の理解にとって重要な麻雀用語としては以下のようなものがある．

- ドラ　得点の加算に繋がる牌．ドラ表示牌と呼ばれる全プレイヤに公開された牌からドラは決定するが，赤ドラと呼ばれる常にドラである牌も存在する．
- チョンボ　重大なルール違反をすること．他プレイヤに決まった得点を支払うことが一般的である．
- 聴牌（テンパイ）　あがりに必要な牌が残り一枚になること．また，そのような手牌を表す．
- フリテン　あがりの形をなすのに必要な最後の一つの牌が自分の捨て牌にあるような手牌構成，またそのような状態のこと．フリテン時はロンあがりができない．

図 2.1　麻雀で用いられる牌．

nn

第 3 章

先行研究

この章では，はじめに強化学習について簡単に説明した後，強化学習を用いた主要な麻雀プレイヤである爆打と Suphx について述べる．また，麻雀においてニューラルネットを用いた方策関数や価値関数の学習を行なった先行研究についても述べる．

3.1 強化学習

この節では強化学習の基本的な説明を行うとともに，今後出てくる用語について説明する．強化学習も複雑な概念のため，ここでは強化学習自体の説明は最小限に留めて重要な用語の説明に話を絞る．

強化学習は，数値化された報酬を最大化するために学習者が何をすべきかを学習するモデルである．学習者はある状態 $s \in \mathcal{S}$ においてある行動 $a \in \mathcal{A}(s)$ を取る．このとき $\mathcal{S}$ は状態集合，$\mathcal{A}(s)$ は s において選ぶことのできる行動の集合である．行動後には次の状態 s' に移行し，報酬 r を受け取る．これを繰り返し行い，終端状態までに獲得した累積報酬を最大化するような行動を学習するのが強化学習の大まかな枠組みとなる．通常の教師あり学習とは異なり，学習者はある状態に対する行動として明確な教師データは与えられない．どの行動を取ればより多くの報酬を得られるかを学習者自身が見つけ出す必要がある．

時刻 t で取られた行動に対して，時刻 $t+1$ にどのように遷移するかを考えるとき，一般的な場合では次状態 s' と報酬 r は，過去の全ての事象の可能な値 $a_t, s_t, r_t, a_{t-1}, s_{t-1}, r_{t-1}, ..., a_0, s_0$ に依存する．そのため遷移確率 Pr は式 (3.1) のようになる．

$$\Pr(s_{t+1} = s', r_{t+1} = r | a_t, s_t, r_t, a_{t-1}, s_{t-1}, r_{t-1}, ..., a_0, s_0) \tag{3.1}$$

一方，時刻 $t+1$ における遷移は時刻 t における状態と行動のみに依存することもあり，ことのき遷移確率 Pr は式 (3.2) のようになる．

$$\Pr(s_{t+1} = s', r_{t+1} = r | a_t, s_t) \tag{3.2}$$

式 (3.1) と式 (3.2) が等しいときにこの環境はマルコフ性を満たすという．マルコフ性を満たす

強化学習タスクは**マルコフ決定過程**と呼ばれる．マルコフ性が成り立つ場合に対して展開された様々な理論は，厳密にはマルコフ性の成り立たない状態を持つタスクに対してもうまく適用できることがあり，より複雑で現実的な非マルコフ性の場合に拡張するための基礎となる．麻雀のように，プレイヤに全ての状態が見えていないタスクの場合には，部分観測マルコフ決定過程としてモデル化することもある．

強化学習の重要な構成要素として，**収益**（return）と**方策**（policy），**価値**がある．収益 G とは，その学習者がその状態を起点として将来にわたって獲得できる（と予想される）報酬を表す．単純な場合では今後得られる報酬の和，つまり r_k を時刻 k における報酬，t を現在の時刻，T を最終時刻として，$G = \sum_{k=t}^{T} r_k$ となるが，割引という概念を考慮に入れる場合もあり，この時は割引率を γ として $G = \sum_{k=t}^{T} \gamma^{k-t} r_k$ となる．割引率には 1 未満の値を設定することで，将来の報酬を小さく見積もる効果がある．方策は，ある状態での学習者のふるまいを定義するもので，ある状態からその状態のときに取るべき行動を出力する関数で表すことができる．また状態の価値とは収益の期待値である．つまりその後に続きそうな状態群とそれらの状態群で得られそうな報酬を考慮にいれた上での長期的な望ましさを示す．強化学習においてはこの価値を正確に求めることで学習者の目標を達成することができる．価値関数とは状態（と行動）から価値を出す関数である．ある方策 π に基づいて状態 s から行動した時に得られる価値を $V_\pi(s)$ と表記し，これを状態価値関数という．また，ある方策 π に基づいて状態 s から行動 a を取った時に得られる価値を $Q_\pi(s,a)$ と表記し，これを状態行動価値関数という．なおこれらは，方策 π を省略して $V(s)$ や $Q(s,a)$ と表記することもある．また最適状態価値関数 $V^*(s)$ は式 (3.3) で定義できる．

$$V^*(s) = \max_\pi V^\pi(s) \tag{3.3}$$

3.2 強化学習を用いた主要な麻雀プレイヤ

本節では強化学習を用いた主要な麻雀プレイヤである爆打と Suphx についてそれぞれ述べる．ここで言う「麻雀プレイヤ」とは自分の得点状況や手牌，捨て牌などの現在の情報を入力として，その時に取る行動を出力するものである．

3.2.1 爆打

水上ら [13] [14] [15] [16] は麻雀プレイヤを複数作成しているが，爆打はその提案手法を用いて作られた麻雀プレイヤの総称である．本項では水上らが強化学習を使用して麻雀プレイヤを作成した手法 [15] について述べる．この手法では，人間の牌譜を用いた教師あり学習により，手牌のみを入力としてその時の切る牌を出力する一人麻雀プレイヤ [13] を使用する．このプレイヤは与えられた手牌から一つ牌を切った時の手牌を想定し，その手牌から抽出される特徴量 $\mathbf{x}$ と重みベクトル $\mathbf{w}$ の内積によって評価値を計算する．これを全ての牌について行い，一番評価値が高い行動を出力する．具体的な評価値 $f(\mathbf{x}, \mathbf{w})$ は式 (3.4) で表される．

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} = \sum_{i=1}^{n} w_i x_i \tag{3.4}$$

ここで x_i と w_i は特徴量と重みベクトルの i 番目の要素を表し，ベクトルの次元数を n とする．また $\mathbf{x}$ と $\mathbf{w}$ はどちらも列ベクトルとする．この一人麻雀プレイヤは鳴ける局面のときに鳴かなかった場合の評価値と，鳴いて手牌から一枚切った手牌の評価値を比較するという方法で，鳴ける局面についても行動を出力することができる．様々な局面を生成するために一局の間に一度だけランダムに行動するなどの改善を加えたプレイヤ同士を対局させて牌譜を生成した．それらを用いて，手牌から和了時の翻数を予測するモデルを強化学習により構築している．このモデルの出力と期待最終順位を用いて点数状況を考慮する麻雀プレイヤは，式 (3.5) の値 $Score(\mathbf{x})$ に基づいて行動する．

$$Score(\mathbf{x}) = \sum_{p \in players} \sum_{i=1}^{4} \frac{P(i|\mathbf{x}) EFR(y_0 + t(i), y_p - t(i))}{4} \tag{3.5}$$

ここで $players$ は全プレイヤの集合を，$P(i|\mathbf{x})$ は特徴量 $\mathbf{x}$ で表される手牌の時に i 翻で和了する確率を，y_0 は自分の現在の点数を，$t(i)$ は i 翻で和了した時の点数を，y_p はプレイヤ p の現在の点数を，$EFR(y_0 + t(i), y_p - t(i))$ は自分が y_0 点で $t(i)$ をプレイヤ p からあがった時の期待最終順位 (Expected Final Rank) を表す．なお $P(i|\mathbf{x})$ について $i = 0$ ではあがれないという場合を表し，4 翻以上のあがりは $i = 4$ に含める．このプレイヤは水上らが以前に作成した麻雀プレイヤ [14] には負け越している．この原因としては，以前の麻雀プレイヤと比べて和了率が 0.01 ほど下がっており，単純な牌効率が悪化した可能性を指摘している．この手法では手牌を $n = 6,661,309$ 次元の特徴量に変換して入力として用いているが，具体的には「役牌の刻子の数」「向聴数」など自身らの麻雀に対する知識を利用して手作業で構築しているため，この特徴量は手生成の特徴量であると言える．

3.2.2 Suphx

Suphx [4] は Microsoft Research Asia が 2019 年 8 月に発表した麻雀プレイヤで，2019 年 6 月にオンラインゲームの天鳳において人間以外で初めて十段を達成した．天鳳の牌譜をもとに教師あり学習を行い，自己対戦による強化学習でモデルを改善している．

Suphx は打牌モデル，リーチモデル，チーモデル，ポンモデル，カンモデルという五つの CNN モデルにルールベースの和了モデルを加えた六つのモデルで構成される．和了モデルは「和了できる時は和了する．ただし，最終局で和了してもラスになる時はあがらない.」という単純なモデルであるが，残りの五つのモデルでは，教師あり学習，強化学習，チューニングの三つの学習ステップを経ている．

教師あり学習においては，各五つのモデルにおいて天鳳の牌譜から 4000 万〜1 億局面を使用し，手牌や得点などの現在の状況を入力としてその局面での行動を学習する．次に，得られたモデルを

元に自己対戦による強化学習でモデルの改善を行う．最後に，このモデルでオンライン対戦を行う時には実際に配られた手牌にモデルをチューニングする run-time policy adaptation と呼ばれる手法を用いた上で行動を決めている．

なお，強化学習時には

1. policy gradient algorithm
2. global reward prediction
3. oracle guiding

という三つの Suphx 独自に基づいた学習を行っている．

1 の policy gradient algorithm では，目的関数に方策のエントロピー正則化項を加えることで方策を適切に学習できるように調整する．Suphx は分散型の強化学習を用いており，式 (3.6) のように古いパラメータから得られた軌跡の重みづけをする importance sampling を適用している．

$$\mathcal{L}(\theta) = E_{s,a\sim\pi_{\theta'}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta'}(a|s)}A^{\pi_\theta}(s,a)\right] \tag{3.6}$$

ここで θ は最新の方策のパラメータ，θ' は古い方策のパラメータ， $A^{\pi_\theta}(s,a)$ はある方策 π で状態 s 行動 a における advantage を表す．advantage とは $Q(s,a) - V(s)$，つまり状態 s の価値に依存しない行動 a の純粋な価値を表すものである．

しかし，方策のエントロピーが小さすぎるときは強化学習はすぐに収束して，自己対戦が方策の改善をしなくなる．また方策のエントロピーが大きすぎるとき強化学習は不安定で，学習した方策の分散が大きくなる．そのため式 (3.7) のように目的関数に方策のエントロピー項を加えている．

$$\nabla_\theta J(\pi_\theta) = E_{s,a\sim\pi_{\theta'}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta'}(a|s)}\nabla_\theta \log \pi_\theta(a|s)A^{\pi_\theta}(s,a)\right] + \alpha\nabla_\theta H(\pi_\theta) \tag{3.7}$$

ここで $J(\pi_\theta)$ は方策 π に従った時の価値の期待値，$H(\pi_\theta)$ は π_θ のエントロピー， α は 0 以上の値を取る係数である．

2 の global reward prediction について，麻雀においては自分から振り込んで得点を少し減らしてでも局を進めた方が良いこともあり，局ごとの収支が必ずしも強化学習の良い報酬とはならない．そこで強化学習における良い報酬を出力するために，現在の局数やそれまでの累積得点といったある局での情報を表す特徴ベクトルを入力として最終結果を予測する Global Reward Predictor Φ を作る．Suphx では Φ に 2 層の Gated Recurrent Unit（GRU）と 2 層の全結合層を足したモデルを採用している．式 (3.8) のように実際に得られた報酬と予測値の二乗誤差を最小化するように天鳳の牌譜から学習する．

$$\min \frac{1}{N}\sum_{i=1}^{N}\frac{1}{K_i}\sum_{j=1}^{K_i}\left(\Phi(x_i^1, ..., x_i^j) - R_i\right)^2 \tag{3.8}$$

ここで K_i は i 番目の局の局数，R_i は i 番目のゲームの報酬を表す．k 局目の特徴ベクトルを

x^k として $\Phi(x^k) - \Phi(x^{k-1})$ を強化学習の報酬に使うということが global reward prediction という手法である．

3 の oracle guiding について，麻雀には山や相手の手牌など，プレイヤからは見えない情報がたくさん存在する．そのようなゲームの全ての情報にアクセスできる oracle agent を取り入れて，強化学習の学習スピードを上げるという手法が oracle guiding である．強化学習のはじめではプレイヤは oracle agent であるが，全ての情報にアクセスできない normal agent に徐々に近づけながら学習を進める．目的関数は式 (3.9) で表される．

$$\mathcal{L}(\theta) = E_{s,a\sim\pi_{\theta'}}\left[\frac{\pi_\theta(a|[x_n(s),\delta_t x_o(s)])}{\pi_{\theta'}(a|[x_n(s),\delta_t x_o(s)])} A^{\pi_\theta}([x_n(s),\delta_t x_o(s)],a)\right] \tag{3.9}$$

これは式 (3.6) の状態 s を $[x_n(s),\delta_t x_o(s)]$ に取り替えたものである．$x_n(s)$ は通常の特徴ベクトル，$x_o(s)$ は追加の完全な特徴ベクトル，δ_t は t 回目のイテレーションにおける dropout matrix で，その要素はベルヌーイ変数 $P(\delta(i,j)) = \gamma_t)$ に従う．γ_t は 1 から 0 に徐々に減らしていき，0 になった時に追加の完全な特徴ベクトルは消え，oracle agent は normal agent になる．

3.3 方策関数の学習を行なった先行研究

本節では，麻雀においてニューラルネットワークを用いた方策関数の学習を行なった築地らの研究と Gao らの研究について述べる．

3.3.1 築地ら

築地ら [7] は，麻雀の特徴を自動抽出して捨て牌を自動選択する学習を行うことを目的として，麻雀用の学習用 CNN 構成を提案した．これは四つのサイズの異なるフィルタを用いた二次元畳み込み層，プーリング層，全結合層からなるネットワークである．二次元畳み込み層のフィルタ枚数は 32，プーリング層では最大プーリングを行い，全結合層の深さは 1 で，活性化関数には ReLU を用いる．全結合層では選定率が 0.5 であるドロップアウトも行なっている．学習データとして「東風荘」[*1]の牌譜 40000 局面を用い，テストデータとの一致率が 53.93% になった．この研究では面子に関する特徴を画像的に表現できる手牌の入力方法も提案している．この入力は 34 行 5 列，要素は 0 or 1 の二値の行列 M で手牌を表現するもので，種類 i の牌が j 枚あった時，$M_{i,j} = 1$ でそれ以外は全て 0 とするというデータ構造になる．このモデルの出力は各牌 34 種に対応した，捨て牌となる確率である．

3.3.2 Gao ら

Gao ら [8] は深層畳み込みニューラルネットワークを用いて不完全情報ゲームである麻雀の教師あり学習を行った．手牌から捨てる牌を出力するモデルにおいては，「二次元畳み込み層，Batch

[*1] `http://mj.giganet.net` 現在はサービスが終了している．

Normalization，ドロップアウト」というネットワークを三つ重ねて，最後に全結合層を入れるモデルを使用している．二次元畳み込み層において，カーネルサイズは 5×2 で，フィルターの枚数は 100，ドロップアウトの選定率は 0.5，全結合層のユニット数は 35 で活性化関数には softmax 関数を用いている．学習データとして「天鳳」の牌譜を用いており，accuracy として 68.8% を達成している．またこの研究でもニューラルネットの構造の提案だけでなく，入力データである手牌の表現方法も紹介もしている．この入力は 34 行 4 列，要素は 0 or 1 の二値の行列 M で手牌を表現するもので，種類 i の牌が k 枚あった時，$j < k$ について，$M_{i,j} = 1$ でそれ以外は全て 0 とするというデータ構造になる．このモデルの出力も各牌 34 種に対応した，捨て牌となる確率である．

第 4 章

ミニ麻雀における実験

この章ではミニ麻雀を用いた教師あり学習の方策のモデル評価の実験について述べる．教師が適切ならばニューラルネットワークで切る牌の選択が最適なものに迫る方策関数を作成可能なこと，またニューラルネットワークのモデルにより学習で得られる方策関数の質に違いがあることを示すのがこの実験での目的となる．また手牌の表現方法によって学習速度に差が出ると考えられるため，適切な手牌の表現方法を見つけるのも目的となる．

はじめにミニ麻雀のルールについて説明し，次に各手牌における割引収益の期待値を求める Value Iteration を行ってその結果について述べる．最後に，得られた期待値を元に教師あり学習を行いニューラルネットワークのモデル並びに手牌の表現方法について評価し，その考察を行う．

4.1　ミニ麻雀のルール

この節ではミニ麻雀のルールを説明する．ミニ麻雀は著者らが文献 [10] によって提案した麻雀を簡略化したゲームである．役を考慮して打牌する必要があるという麻雀の性質を保持しながらも，使用する牌の種類や手牌の枚数などを減らすことで理論的な最善手が求められるという特徴を持つ．

以下に通常の麻雀との違いを中心にルールの説明をする．

- 使用する牌の種類は萬子のみの九種類．同種の牌の枚数は通常の麻雀と同じで四枚．
- 手牌の枚数は七枚とし，雀頭一つと面子を二つ構成することであがりとする．
- 一人でプレイする．したがって，ツモあがりしか存在しない．
- ポン，チー，カンなどの鳴きはない．
- 捨て牌は山に戻して毎回シャッフルしてから引く．そのためツモ牌は手持ち以外の牌から等確率で引くことになる．例えば手牌に一萬が三枚，二萬が二枚，三萬が一枚もない時，二萬を引く確率は一萬の倍であり，三萬は更にその倍になる．
- ツモの回数に制限はなく，あがるまでゲームを続ける．
- プレイの目的は割引収益の期待値を最大化することとする．

- 役はタンヤオ，一盃口，対々和，チャンタのみを考慮する．役の説明は表 4.1 に示した．役を導入したミニ麻雀を「役ありミニ麻雀」，役を導入しないミニ麻雀を「役なしミニ麻雀」とする．
- 役ありミニ麻雀の場合，あがり時の役の個数を v として，あがった時の得点は 2^v とする[*1]．役なしミニ麻雀の場合は，あがりの形になった際に得点 1 を得る．

表 4.1　ミニ麻雀で用いる役の名前とその説明.

名前	説明
タンヤオ	2 から 8 のみであがりの形を構成
チャンタ	各面子と雀頭に一九字牌を含む
一盃口	同一の順子で二面子を構成する
対々和	同一の刻子で二面子を構成する

このゲームは報酬はあがった時の得点 r のみのエピソードタスク（episodic task）に対応する．スタートから n 手であがった時の割引収益 G は割引率を γ として，

$$G = \gamma^n r \tag{4.1}$$

となる．本実験においては割引率 γ は 0.9 とする．

一般にエピソードタスクの時には割引率を 1 に設定することもあるが，高い役であがることと早くあがることという相反する目的を達成することは麻雀の重要な特徴であり，割引率を 1 とした時には前者のみを達成することが目的となるため，割引率は 1 未満に設定している．

4.2　Value Iteration

ミニ麻雀においては，現在の巡目や捨牌の記録は最善手を求めるにあたって不要である．手牌から一枚捨てた時の手牌の組み合わせから，最適な戦略を取った時の割引収益の期待値を求めることができる．一枚切った時の手牌の状態集合を $\mathcal{S}$ として，七枚の手牌の全状態数 $|\mathcal{S}|$ は 6,030 通りある[*2]．Sutton ら [17] において「Value Iteration」として紹介されているアルゴリズムにより，最適な戦略を取った時の各状態に対応する割引収益の期待値 $V^*(s)$ を求めた．具体的には Algorithm 1 である．

ここで，$p(s', r|s, a)$ は状態 s において行動 a を取り，報酬 r を受け取って状態 s' に遷移する確率を表す．また $terminal$ は終端状態を表す．

ここで得られた結果について少し考察する．全ての手牌の価値の平均値 $E_{s\in\mathcal{S}}[V^*(s)]$ は 1.484 である．あがっていない状態の八枚の手牌の組み合わせは 10,389 通りあり，役ありミニ麻雀

[*1] ただしタンヤオとチャンタ，一盃口と対々和のように複合しない役があるため，得点の最大値は 4 である．

[*2] 1 から 9 までを 9 から 1 までに置き換える対称性を利用して組み合わせ数を減らすことが可能だが，この数字はそれを考慮していない．

Algorithm 1 Value Iteration

Algorithm parameters: a small threshold $\theta = 10^{-6}$, discount rate $\gamma = 0.9$
Initialize $V(s)$, for all $s \in \mathcal{S}$, arbitrarily except that $V(terminal) = 0$
loop
  $\Delta \leftarrow 0$
  for each $s \in \mathcal{S}$ **do**
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
    $\Delta \leftarrow max(\Delta, |v - V(s)|)$
  end for
  if $\Delta < \theta$ **then**
    break
  end if
end loop

($r = 2^v$) と役なしミニ麻雀 ($r = 1$) で，最善手が変わった手牌はそのうち 6570 個ある．最善手が変わる手牌の中のうち，役ありミニ麻雀での最善手と次善手の価値の差が最も大きい手牌の一つは「22778888」である．役ありミニ麻雀での最善手は 8 切り[*3]で価値は 2.462（役なしミニ麻雀での最善手は 7 切り[*4]）であるが，次善手は 2 切りで価値は 1.671 となっている．

同様に，役を導入するかどうかで最善手が変わる手牌の中のうち，役なしミニ麻雀での最善手と次善手の価値の差が最も大きい手牌の一つは「56789999」である．役なしミニ麻雀での最善手は 9 切り[*5]で価値は 0.890（役を導入する時の最善手は 6 切り[*6]）であるが，次善手は 5 切りで価値は 0.741 となっている．これらの手牌の例を図 4.1 に示す．

4.3 教師あり学習

この節では，ある手牌が与えられた時に最善の捨牌を求める方策関数となるニューラルネットワークについていくつか評価する．麻雀を行うプログラムには方策関数を学習するもの，価値関数を学習するもの，その両方を学習するものなど存在するが，ここで方策関数のみを対象としたのは，方策関数の方が理論上の最適プレイヤにどのくらい近づいたかが評価しやすいためである．あがっていない状態の八枚の手牌の組み合わせと，牌を捨てた後の割引収益の期待値が最大となる手を 1（複数ある時は最大となる種類数で 1 を割った値），それ以外を 0 とするような教師データを

[*3] 2 と 7 であがることができ，タンヤオと対々和とが確定する．
[*4] 6 と 9 であがることができ，8 切りよりもあがることのできる枚数が倍になる．
[*5] 4，5，7，8 という四種類の牌であがることができる．
[*6] 5 でしかあがれないものの，7 や 8 引きで一盃口とチャンタが確定する待ちになる．

図 4.1　役を導入するかしないかで最善手が変わる手牌のうち，役ありミニ麻雀での最善手と次善手の価値の差が最も大きい手牌（上）と役なしミニ麻雀での最善手と次善手の価値の差が最も大きい手牌（下）．

作成して教師あり学習を行う．

ここで，入力手牌の表現方法には複数考えられ，その表現方法によって学習速度に差が出ると考えられる．本研究では以下の三種類の表現方法を検討した．

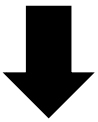
一つ目は築地らの提案した入力をミニ麻雀に適用したものである．提案手法においては全ての種類の牌を使用していたため 34 行 5 列の行列で手牌を表現しているが，この表現方法をミニ麻雀に適用した場合，9 行 5 列，要素は 0 or 1 の二値の行列 M で手牌を表現し，種類 i の牌が j 枚あった時，$M_{i,j} = 1$ でそれ以外は全て 0 とする．図 4.2 に，例として手牌が「11244999」であった時の表し方を示した．本研究ではこの手牌の表現方法を「手牌モード 1」と名付ける．

二つ目は Gao らの提案した入力をミニ麻雀に適用したもので，提案手法においては全ての種類の牌を使用していたため 34 行 4 列の行列で手牌を表現しているが，この表現方法をミニ麻雀に適用した場合，9 行 4 列，要素は 0 or 1 の二値の行列 M で手牌を表現するもので，種類 i の牌が k 枚あった時，$j < k$ について，$M_{i,j} = 1$ でそれ以外は全て 0 とする．図 4.3 に，例として手牌が「11244999」であった時の表し方を示した．本研究ではこの手牌の表現方法を「手牌モード 2」と名付ける．

三つ目は 8 行 9 列，要素は 0 or 1 の二値の行列 M で手牌を表現するもので，手牌を種類によってソートした後で，i 枚目の牌の種類が j の時，$M_{i,j} = 1$ でそれ以外は全て 0 とする．図 4.4 に，例として手牌が「11244999」であった時の表し方を示す．本研究ではこの手牌の表現方法を「手牌モード 3」と名付ける．

この章における実験の設定は以下の通りである．

- 10,389 通りのうち，75 % を訓練データとし，25 % をテストデータとして用いる．
- 訓練データを一回ずつ学習させるのを 1 エポックとし， エポック数は 1000 回とする．
- loss は categorical cross entropy とする．



	0枚	1枚	2枚	3枚	4枚
1m	[0,	0,	1,	0,	0]
2m	[0,	1,	0,	0,	0]
3m	[1,	0,	0,	0,	0]
4m	[0,	0,	1,	0,	0]
5m	[1,	0,	0,	0,	0]
6m	[1,	0,	0,	0,	0]
7m	[1,	0,	0,	0,	0]
8m	[1,	0,	0,	0,	0]
9m	[0,	0,	0,	1,	0]

図 4.2　手牌が「11244999」の時の手牌モード 1 の表現例.

4.4　提案するモデルにおける教師あり学習

方策関数に適切なネットワーク構造を見つけるため，以下の三つの構造を考える.

1. 多層パーセプトロン
2. conv1d + Dense
3. conv2d + Dense

これらのネットワークは先行研究を元に作ったものである．はじめにハイパーパラメータ自動最適化ツールである Optuna について説明したあと，それぞれのネットワークについて説明する.

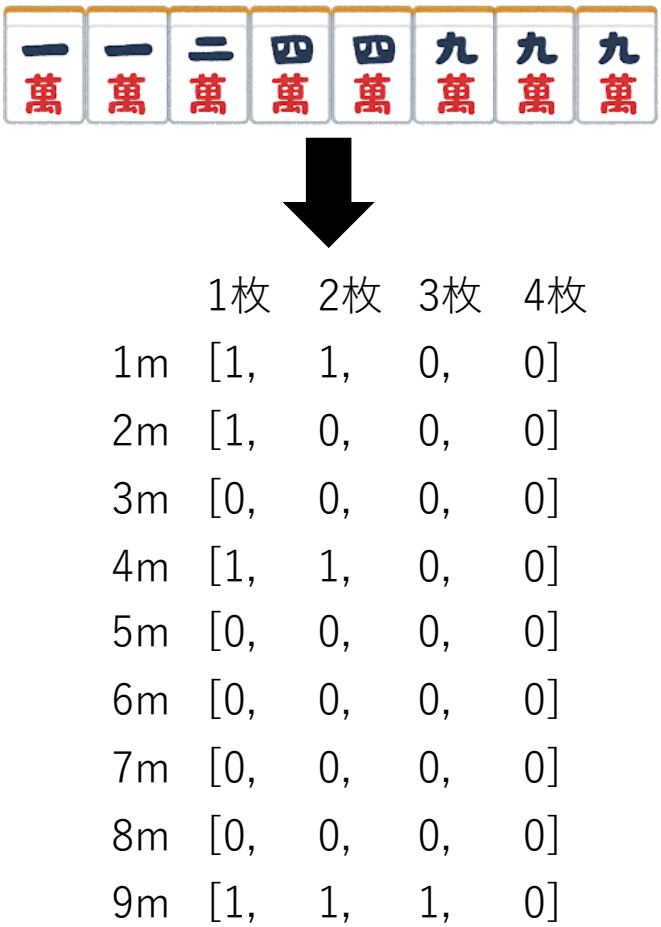


図 4.3　手牌が「11244999」の時の手牌モード 2 の表現例.

4.4.1　Optuna

Optuna [9] とは，秋葉拓哉らが作成し，2018 年 12 月[*7]に公開されたハイパーパラメータ自動最適化ツールである．

ハイパーパラメータとは，機械学習においてはアルゴリズムによって最適化できない，もしくはしないパラメータのことであり，学習前に人があらかじめ決定しておく必要がある．具体的にはニューラルネットワークの層の数やユニットの数などが該当する．学習時には適切に設定されないと，学習がうまく行われないこともあり，非常に重要なパラメータである．

Optuna においては次の試行で試すべきハイパーパラメータの値を決めるために，それまでに完了している試行の履歴を用いる．具体的には，Tree-structured Parzen Estimator [18] というベイズ最適化アルゴリズムの一種を用いる．そこまでで完了している試行の履歴に基づき，有望そうな

[*7] https://tech.preferred.jp/ja/blog/optuna-release/

	1m	2m	3m		8m	9m
1枚目	[1,	0,	0,	…	0,	0]
2枚目	[1,	0,	0,	…	0,	0]
3枚目	[0,	1,	0,	…	0,	0]
4枚目	[0,	0,	0,	…	0,	0]
5枚目	[0,	0,	0,	…	0,	0]
6枚目	[0,	0,	0,	…	0,	1]
7枚目	[0,	0,	0,	…	0,	1]
8枚目	[0,	0,	0,	…	0,	1]

図 4.4　手牌が「11244999」の時の手牌モード 3 の表現例.

領域を推定し，その領域の値を実際に試すということを繰り返す.

ハイパーパラメータはグリッドサーチなどによって値を決めることもあるが，ハイパーパラメータの候補が多いと時間がかかるため，本研究では Optuna を用いた.

4.4.2　多層パーセプトロン

「多層パーセプトロン」モデルについて，これは全結合層を主に使用して作られたモデルであり，Optuna を用いて最適なパラメータを求めた．全結合層の中間層の数 i は 1 から 3，それぞれのユニットの数 u_j $(j \in \{1, 2, \ldots, i\})$ は 50 から 1000 の範囲から取るものとし，Optuna における試行回数は 200 回とした．この時のネットワークを図 4.5 に示している.

この条件において最も良い値が得られたネットワークの構造は以下である.

- $i = 2$，つまり中間層が 2 層の 4 層 MLP で，はじめにデータを一次元配列に直す.
- $u_1 = 302$，つまり中間層 1 層目のユニットの数は 302 である.
- $u_2 = 125$，つまり中間層 2 層目のユニットの数は 125 である.
- 出力層のユニットの数は行動の数，すなわち 9 である.

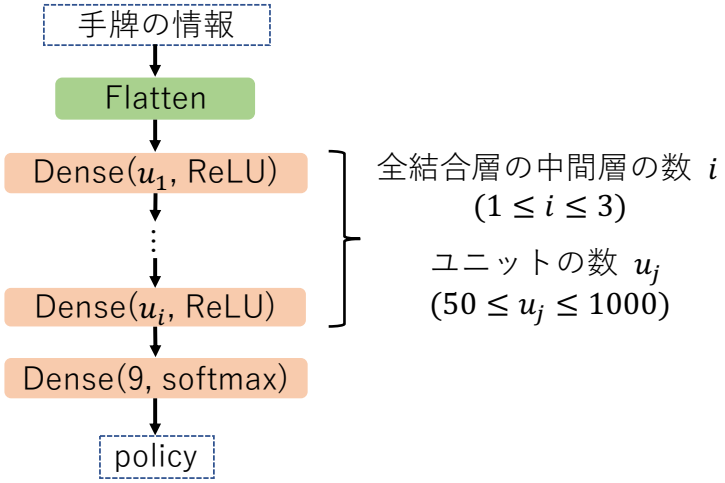


図 4.5　「多層パーセプトロン」モデルにおけるネットワークの構成図.

- 出力層の活性化関数は softmax 関数で，それ以外では ReLU を用いる．

4.4.3　conv1d + Dense

次に「conv1d + Dense」モデルについて，これは一次元畳み込み層と全結合層を主に使用して作られたモデルであり，Optuna を用いて最適なパラメータを求めた．一次元畳み込み層の数 i は 1 から 4，それぞれのフィルタの数 f_j $(j \in \{1, 2, \ldots, i\})$ は 8 から 64，全結合層の中間層の数 k は 1 か 2，それぞれのユニットの数 u_l $(l \in \{1, \ldots, k\})$ は 50 から 1000 の範囲から取るものとし，Optuna における試行回数は 200 回とした．なお「一次元畳み込み層」の軸の方向は数牌の枚数の方向ではなく種類ごとの方向である．この時のネットワークの構成図を図 4.6 に示している．

この条件においてもっとも良い値が得られたネットワークの構造は以下である．

- $i = 2$ かつ $k = 1$，つまりはじめの 2 層は一次元畳み込み層とし，その後にデータを一次元配列に変更してから全結合層の中間層を 1 層入れる．
- 一次元畳み込み層におけるカーネルサイズは 3，padding は SAME とし，活性化関数には ReLU を用いる．
- $f_1 = 56$ かつ $f_2 = 64$，つまり一次元畳み込み層におけるフィルタの数は，はじめの層からそれぞれ 56，64 である．
- $u_1 = 711$，つまり全結合層の 1 層目のユニットの数は 711 である．
- 全結合層の出力層のユニットの数は行動の数，すなわち 9 である．
- 全結合層の出力層の活性化関数は softmax 関数で，それ以外では ReLU を用いる．

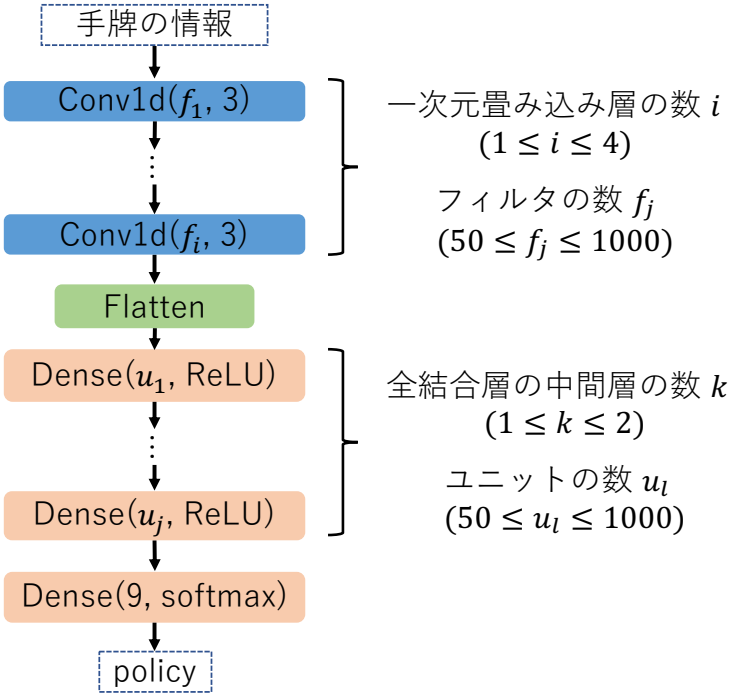


図 4.6　「conv1d + Dense」におけるネットワークの構成図.

4.4.4　conv2d + Dense

次に「conv2d + Dense」モデルについて，これは二次元畳み込み層と全結合層を主に使用して作られたモデルであり，Optuna を用いて最適なパラメータを求めた．二次元畳み込み層の数 i は 1 から 4，それぞれのフィルタの数 f_j $(j \in \{1, 2, \ldots, i\})$ は 8 から 64，全結合層の中間層の数 k は 1 か 2，それぞれのユニットの数 u_l $(l \in \{1, \ldots, k\})$ は 50 から 1000 の範囲から取るものとし，Optuna における試行回数は 200 回とした．この時のネットワークの構成図を図 4.7 に示している．

この条件においてもっとも良い値が得られたネットワークの構造は以下である．

- $i = 2$ かつ $k = 2$，つまりはじめの 2 層は二次元畳み込み層とし，データを一次元配列に変更してから全結合層の中間層を 2 層入れる．
- 二次元畳み込み層におけるカーネルサイズは 3, padding は SAME，活性化関数には ReLU を用いる．
- $f_1 = 64$ かつ $f_2 = 43$，つまり二次元畳み込み層におけるフィルタの数は，はじめの層からそれぞれ 64，43 である．
- $u_1 = 925$ かつ $u_2 = 492$，つまり全結合層のユニットの数は 1 層目が 925，2 層目が 492 である．

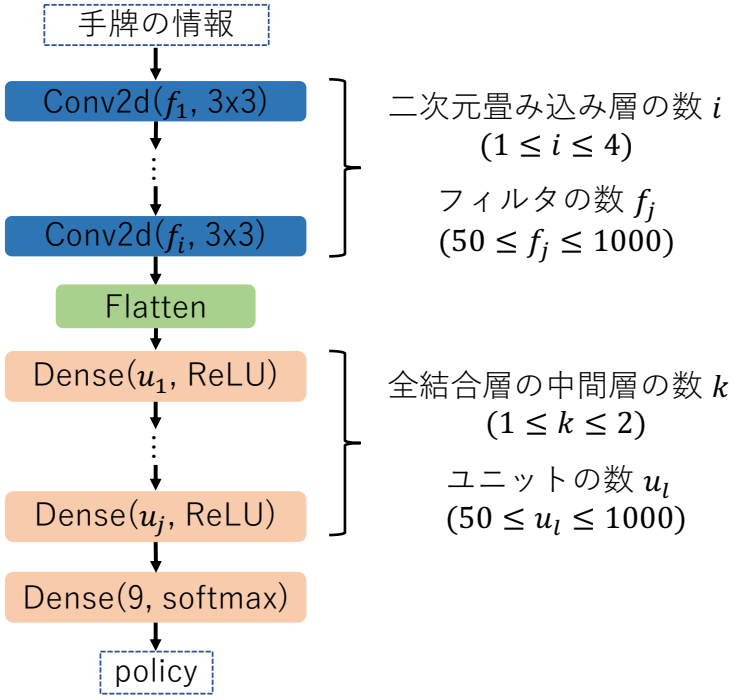


図 4.7　「conv2d + Dense」におけるネットワークの構成図.

- 全結合層の出力層のユニットの数は行動の数，すなわち 9 である.
- 全結合層の出力層の活性化関数は softmax 関数で，それ以外では ReLU を用いる.

次にベースラインとして，築地らの実験 [7] と Gao らの実験 [8] を参考にしたモデルを作成した．便宜上前者を「築地モデル」，後者を「Gao モデル」と名づける.

築地らの論文 [7] の実験では，入力として手牌モード 2 を使用した手牌を入れ，出力としては各牌（論文と同じ 34 種）に対応した捨て牌となる確率を出すモデルを使用した．現在用いている牌は 9 種類だが，論文では 34 種類すべての牌を用いており，原論文と全く同じ形を採用している．なお，論文ではカーネルサイズとして 9×5 よりも大きいものも使用していたが，今回の実験では萬子以外の牌を使用していないことから，二次元畳み込み層におけるカーネルサイズは 3×5 と 9×5 のみを使用している．フィルタも原論文では四種類用いているが，三種類は複数の牌種を使用していることが前提のフィルタであるため，一種類のフィルタしか用いていない.

Gao らの論文 [8] の実験では，入力として手牌モード 3（ただし行数は論文と同じ 34）を使用した手牌を入れ，出力としては各牌（論文と同じ 34 種）に対応した捨て牌となる確率を出すモデルを使用した．なお，Batch normalization を行うテンソルの軸を適切と思われるものに修正（通常 Batch normalization は channel の軸に沿って行う）し，出力の次元も牌の種類数の 34 種に修正している．また原論文においては手牌から捨てる牌を出力するモデルだけでなく，リーチを判定するモデルや鳴きについて判定するモデルも作成しているが，ミニ麻雀においてはどちらも考慮しないためこれらのモデルは作成していない.

はじめに学習が進んでいるかを確認するために，役なしミニ麻雀（$r = 1$）で教師あり学習を行い，正解率を調べた．ここでの「正解」とは，出力された確率がもっとも高い牌がその手牌における最善手に含まれていることであり，八枚の手牌の中であがっていない全ての手牌において「正解」の割合を調べたものが「正解率」である．結果を表 4.2 に示す．表中の x は原論文との入力の形が違うために行なっていない実験である．

表 4.2　各手牌モード，モデルにおける正解率（教師あり学習，役なしミニ麻雀）．

名前	手牌モード 1	手牌モード 2	手牌モード 3
多層パーセプトロン	**0.848**	0.847	0.793
conv1d + Dense	**0.855**	0.847	0.822
conv2d + Dense	0.846	**0.853**	0.828
築地モデル	0.807	x	x
Gao モデル	x	0.828	x

手牌モード 1 や手牌モード 2 において，先行研究よりも約 2 から 4 ポイント高い正解率が出た．これは Optuna による自動最適化ツールを用いたことにより，先行研究のモデルよりも教師あり学習に適したモデルを用いることができたからであると考えられる．また手牌モード 3 は他の手牌モードに比べて値が悪くなっている．これは手牌モード 1 や 2 の方が，手牌を画像として表現しやすくなっていることが原因として考えられる．

同様に，役ありミニ麻雀（$r = 2^v$，v は成立した役の数）でも Value Iteration を行い，理論値を出した上でその状態の価値を学習できているかを調べた．正解率の結果を表 4.3 に示す．

表 4.3　各手牌モード，モデルにおける正解率（教師あり学習，役ありミニ麻雀）．

名前	手牌モード 1	手牌モード 2	手牌モード 3
多層パーセプトロン	0.854	**0.888**	0.881
conv1d + Dense	0.861	0.869	**0.886**
conv2d + Dense	0.873	**0.889**	**0.889**

手牌モード 1 の正解率がやや低くなっているものの，役ありという条件においても役なしと同程度の正解率をどの手牌モードにおいても得ることができ，適切にポリシー関数を学習できたと考えられる．

本研究では手牌の入力および手牌から役を考慮して有効な打牌を選択するのに必要なネットワークモデルを検討するために，ミニ麻雀を提案した．一方で，ミニ麻雀は「他のプレイヤからのロンあがり，他のプレイヤへの振り込みを考慮した打牌選択」や「他のプレイヤーとの得点差，順位を考慮した手作り」の要素は含んでいない．ミニ麻雀にこれらの要素を入れたルールも検討したが，その時に適切なゲームバランスでないと意味がないので，次章では，それらの要素を含んだ小さい

麻雀のバリエーションであるすずめ雀を扱う.

第 5 章

すずめ雀における実験

この章ではすずめ雀を対象とした実験について述べる．ミニ麻雀で得られたモデルや手牌の表現方法を踏まえて，ミニ麻雀よりも扱う牌の種類や役が増えてより通常の麻雀に近づいたすずめ雀 [11] を扱い，これに強化学習を適用して，より強いすずめ雀プレイヤ構築するのが目的となる．

はじめはすずめ雀のルールについて，次に一人すずめ雀プレイヤの作成について，強化学習によるすずめ雀プレイヤの作成について述べる．最後に，各局の点数の最大化を目指すのではく，全局を終えたときの平均順位の更なる改善を目的として Global Reward Predictor の作成とそれを用いた強化学習の実験について述べる．

5.1 すずめ雀のルール

この節ではすずめ雀のルールを説明する．すずめ雀は丸田康司，篠崎高広らによって開発された麻雀類似ゲームであり，2018 年 5 月に発売された．発案者の報告によると，2020 年 12 月時点で累計販売数は 15000 個を超えている[*1]．すずめ雀は通常の麻雀とは以下の点で異なっている．

- プレイヤの数は二人から五人とする．
- 各プレイヤの手牌は五枚．
- あがりには面子を二つ作る必要がある．
- 使用する牌は図 5.1 のように索子の九種類と字牌の二種類（發と中）である．同一牌の枚数は四枚で通常の麻雀と同じだが，索子の各牌には一枚ずつ赤牌が存在する．
- 中は全て赤ドラとする．
- ドラはドラ表示牌と同じ牌とする．ただし通常の麻雀と同じように，赤牌とそうでない牌は同じとみなしてドラを定める．
- ポン，チー，カンなどの鳴きはない．
- 役は表 5.1 のものがある．それらの役に加えて手牌で構成する二面子において，順子を構成することで 1 点，刻子を構成することで 2 点を得ることができる．役満成立時には，その役

[*1] https://vjumplay.com/generated/vlog/1056

満の得点が手に入る．

- あがるのに必要な最低得点は 5 点とする．
- 複数プレイヤから同時にロンすることが可能．このとき，牌を捨てたプレイヤーが全ての得点を支払う[*2]．
- フリテンは存在するが，現物のみがフリテンとなる．通常の麻雀ではあがり牌のうち一種類でも自分の捨て牌にあったらロンあがりはできないが，すずめ雀においては自分が捨てていない種類の牌ならロンあがりができる．
- 山牌（ドラ表示牌を除く）を使い切る（二人プレイの時は 16 巡と 1 人，五人プレイの時は 3 巡と 3 人）と流局となり，親が変わる．
- 親があがった時は，上記の役の得点と手牌構成の得点の合計得点にさらに 2 点加算される．ただし，この得点はあがるのに必要な最低得点には含めない．
- ツモあがりは，他のプレイヤで等分とする．ただし端数は切り上げる．
- 各プレイヤは最初に 40 点ずつ持つ．得点がマイナスになってもゲームは続行する．
- 連荘は存在せず，全プレイヤが四回ずつ親を行う（全プレイヤの数を p とするとき，全部で $4 \times p$ 局行う）ことでゲームが終了する．
- チョンボをした時には，自分以外のプレイヤに 2 点ずつ支払って，そのままゲームを続行する．
- すずめ雀の公式ルールでは，自分の得点がマイナスになったらそれ以上支払わなくて良い．しかし本研究では自分の得点がマイナスになっても支払うことにする．

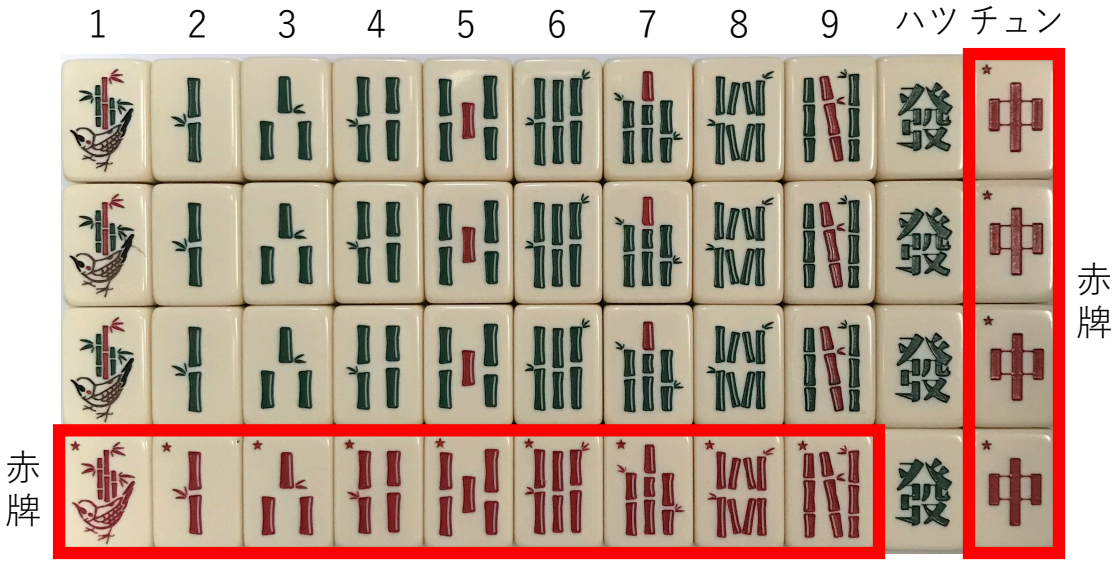


図 5.1　すずめ雀で用いる牌の一覧．

[*2] いわゆる「上家取り」のルールはない

表 5.1　通常役と役満の名前，得られる得点，役の説明.

通常役		
名前	得点	説明
タンヤオ	1	2 から 8 のみで二面子を構成
チャンタ	2	各面子に一九字牌を含む
ドラ	1	一枚につき一点
赤ドラ	1	一枚につき一点
役満		
オールグリーン	10	2, 3, 4, 6, 8, 發のみで二面子を構成
チンヤオ	15	一九字牌のみで二面子を構成
スーパーレッド	20	赤牌のみで二面子を構成

例えば三人対戦で親が図 5.2 のように「3477r7」（r7 は赤牌の 7 を表す）という手牌の時に，「5」をツモったとする．この時の得点は，タンヤオ 1 点，赤ドラ 1 点，順子 1 点（345），刻子 2 点（77r7）で，あがるために必要な 5 点を満たしている．親のあがりの場合は 2 点が加算されるので，あがり時の合計得点は 7 点となる．ツモ時はそれを他の二人のプレイヤで等分して端数は切り上げるため，親は子から 4 点ずつの計 8 点を獲得する．

ツモ

図 5.2　「3477r7」で「5」をツモあがりした時の手牌.

このすずめ雀には，「他のプレイヤからのロンあがり，他のプレイヤへの振り込みを考慮した打牌選択」や「他のプレイヤーとの得点差，順位を考慮した手作り」というミニ麻雀になかった要素が存在するためこのゲームを選択している．

5.2　一人すずめ雀プレイヤの作成

はじめに自分の手牌のみから行動を決定するすずめ雀プレイヤを作成し，牌譜生成やベースラインプレイヤとして使用する．このすずめ雀プレイヤは以下のような条件のもとで，割引収益の期待値が高い牌を切ることができるようなプレイヤである．

- ドラ表示牌と自分の手牌に存在しない牌を引く確率は，どの牌もそれぞれ等確率であるとする．例えば手牌に中が三枚，發が二枚，1 索が一枚もない時，發を引く確率は中を引く確率

の倍であり，1 索を引く確率は更にその倍になる．

- ツモの回数に制限はなく，あがるまで引く．
- あがり時の得点を報酬とし，あがり時以外の報酬は 0.
- 割引率 γ は 0.9 を用いる．

このタスクも報酬はあがった時の得点 r のみのエピソードタスクに対応するため，式 (4.1) と同じように割引収益が定義できる．

このタスクでは現在の巡目や捨て牌に関する情報は最善手を求めるにあたって不要であり，一枚捨てた時の手牌と現在のドラという二つの情報から，最適な戦略を取った時の割引収益の期待値を求めることができる[*3]．一枚切った時の手牌とドラを合わせた状態集合を $\mathcal{S}$ として，全状態数 $|\mathcal{S}|$ は 529647[*4]である．最適な戦略を取ったときの各状態に対応する割引収益の期待値 $V^*(s)$ を Value Iteration [17] により求めた．なおミニ麻雀と同じように $\theta = 10^{-6}$ としている．

ここで得られた結果について少し考察する．全ての状態の価値の平均値 $E_{s\in\mathcal{S}}[V^*(s)]$ は 2.838 である．価値が最も低い手牌の一つは「45678」[*5]で価値が 1.21 となる．価値が最も高い手の一つは「r7r8 中中中」で価値が 10.18 となる．聴牌時で最も価値が低い手の一つは「23789」[*6]で価値が 1.49 となる．非聴牌時で最も価値が高い手の一つは「r3r7 中中中」で価値が 6.95 となる．ここまでに紹介した最も高い（低い）時の価値は全て「中」がドラの時である．これらの例を図 5.3 に示す．

この求めた期待値のテーブルから通常のすずめ雀をプレイできるプレイヤを作成する．具体的には割引収益の期待値が最大となる手を切り，期待値が最大となる手が複数ある時はそのうちの一つをランダムに選択して切り，チョンボ（あがりの最低点数に満たない，フリテンでない）せずにロンあがりができる時は必ずロンあがりするプレイヤである．これ以降ではこのプレイヤを「一人すずめ雀プレイヤ」と呼ぶ．

手牌とドラしか見ない一人すずめ雀プレイヤがどの程度強いかを確かめるために，自分以外のプレイヤをすべてランダムで行動するプレイヤとして対戦させた．手牌から牌を一枚切るという行動を 1step として，二人プレイで 3×10^6 step（対局数にすると 52845 ゲーム 422760 局）分行ったところ，平均順位が 1.00（1 位が 52832 回）であった．五人プレイ時でも 3×10^6 step（対局数にすると 54317 ゲーム 1086340 局）分行ったところ，平均順位が 1.139（1 位が 49292 回）であり，一人すずめ雀プレイヤはランダムプレイヤよりは有意に強くなることが確認できた．またあがり回数は二人プレイ時で 381602 回（ツモあがりが 184165 回，ロンあがりが 197437 回），五人プレイ時で 301255 回（ツモあがりが 40403 回，ロンあがりが 260852 回）であった．二人プレイ時と五人プレイ時のあがり時の巡目とその回数をヒストグラムにしたものを，それぞれ図 5.4 と図 5.5 に

[*3] ロンあがりが存在しないため，自分の捨て牌に関する情報を覚えておく必要はない．

[*4] 1 から 9 までを 9 から 1 までに置き換える対称性を利用して組み合わせ数を減らすことが可能だが，この数字はそれを考慮していない．

[*5] 3，6，9 待ちの聴牌形であるが，タンヤオが付く 3 や 6 でも最低得点に満たないためあがれない．3 や 6 であがるためにはドラや赤ドラが最低でも 2 枚必要である．

[*6] 「r1」（赤ドラの「1」）でのみあがることが可能．

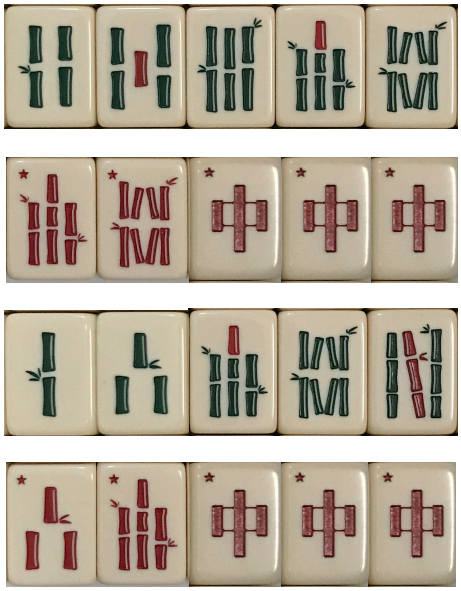


図 5.3　上から順に，価値が最も低い手の「45678」，価値が最も高い手の「r7r8 中中中」，聴牌時で最も価値が低い手の「23789」，非聴牌時で最も価値が高い手の「r3r7 中中中」.

示す．一人すずめ雀プレイヤは二人プレイ時において 9 割近い局を，五人プレイ時において 3 割ほどの局をあがれるということが読み取れる．あがりの平均巡目は二人プレイで 7.58 巡，五人プレイでは 2.66 巡であった．

また，あがり時の平均獲得得点は二人プレイ時で 8.08 点，五人プレイ時で 6.99 点であった．二人プレイ時と五人プレイ時の流局数とあがり回数，その時の獲得得点をヒストグラムで表したものを図 5.6 と図 5.7 に示す．

二人プレイ時では 7 点のあがりが最頻出であるが，五人プレイでは 8 点のあがりが最頻出となっている．これは五人プレイでの 5 点から 8 点のツモあがりが，端数切り上げによって 8 点として換算されることが影響していると考えられる．

ある程度の強さのプレイヤ同士の対戦結果を集めることにより，すずめ雀というゲームの性質を求めることができることを期待して，全員が一人すずめ雀プレイヤであったときも 3×10^6 step 分（二人プレイ時で 70079 ゲーム 560632 局，五人プレイ時で 68058 ゲーム 1361160 局）だけ対戦を行った．この時のあがり回数は，二人プレイ時で 279330 回（ツモあがりが 140430 回，ロンあがりが 138900 回），五人プレイ時で 240471 回（ツモあがりが 38499 回，ロンあがりが 201972 回）であった．あがりの平均順目は二人プレイ時で 5.8 巡，五人プレイ時で 2.43 巡であった．二人プレイ時と五人プレイ時のあがり時の巡目とその回数をヒストグラムにしたものを，それぞれ図 5.8 と図 5.9 に示す．

一人すずめ雀同士の対戦の場合，二人プレイでは最大で 17 巡あるため流局はほとんど起こらないが，五人プレイでは最大で 4 巡しかないこともあり流局は全体の 2 割ほど発生している．

また二人プレイ時では平均獲得得点が 7.69 点，五人プレイ時では 6.96 点であった．二人プレイ

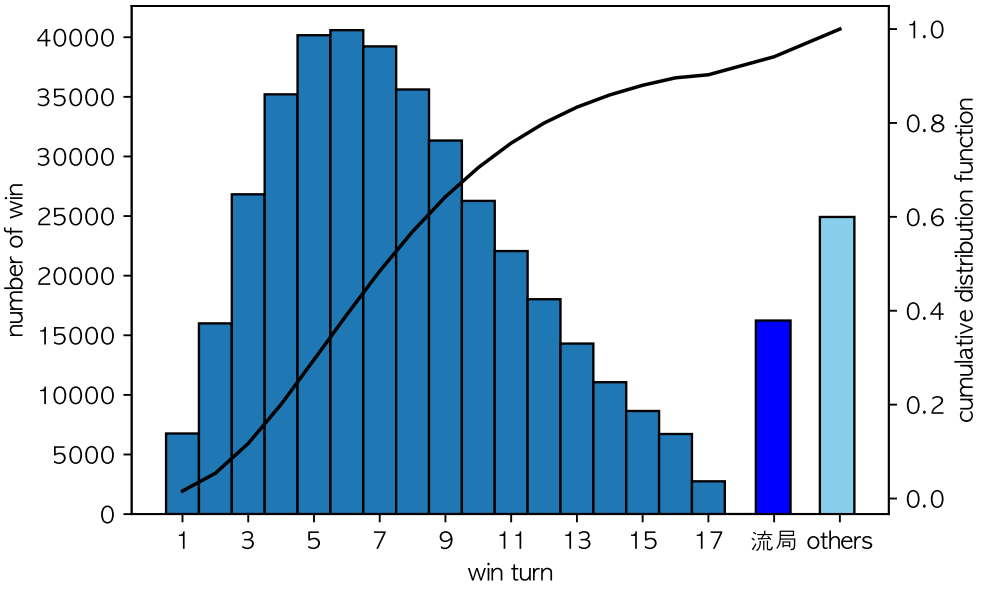


図 5.4　ランダムプレイヤとの二人プレイ対戦時の流局回数，他家のあがり回数（others），該当する順目でのあがり回数（ヒストグラム左軸）．累積分布関数（黒線右軸）．

時と五人プレイ時の流局数とあがり回数，その時の獲得得点をヒストグラムで表したものを図 5.10 と図 5.11 に示す．

図 5.6 と図 5.10，図 5.7 と図 5.11 を比べたとき，対戦相手が強くなった分あがれる回数が減っているが単峰であることや裾の長さなど全体としての山の形は大きく変化していないことが読み取れる．

5.3　強化学習によるすずめ雀プレイヤの作成

この節では強いすずめ雀プレイヤを作成することを目的として，すずめ雀を強化学習で行うための環境を作成し，強化学習を行ってすずめ雀プレイヤを作成する．ここで作成するプレイヤは「現在の状況から何を切るか」のみを実現していて，「あがれる時にどの相手からあがるかを重視してあがりを見逃す」などの戦略はなく，「あがれる時は常にあがる」プレイヤである．また麻雀においてプレイヤが最大化を目指す順位点は，最終順位や最終得点によって決まる．すずめ雀のルールには順位点の記述が無いが麻雀の要素には必要なものために導入する．今回は p 人の対戦で r 位であった時，順位点は $1 + 2 \times (1 - r)/(p - 1)$ とした[*7]．報酬としては，現在の順位を順位点に変換したものと局終了時の得点を 1 から-1 に正規化したものを平均した値を使用した．強化学習アルゴリズムとしては Suphx でも使われている PPO [19] を使用した．モデルには，強化学習用

[*7] 1 位の順位点を 1，ラスの順位点を-1 として残りの順位は均等になるように設定した．

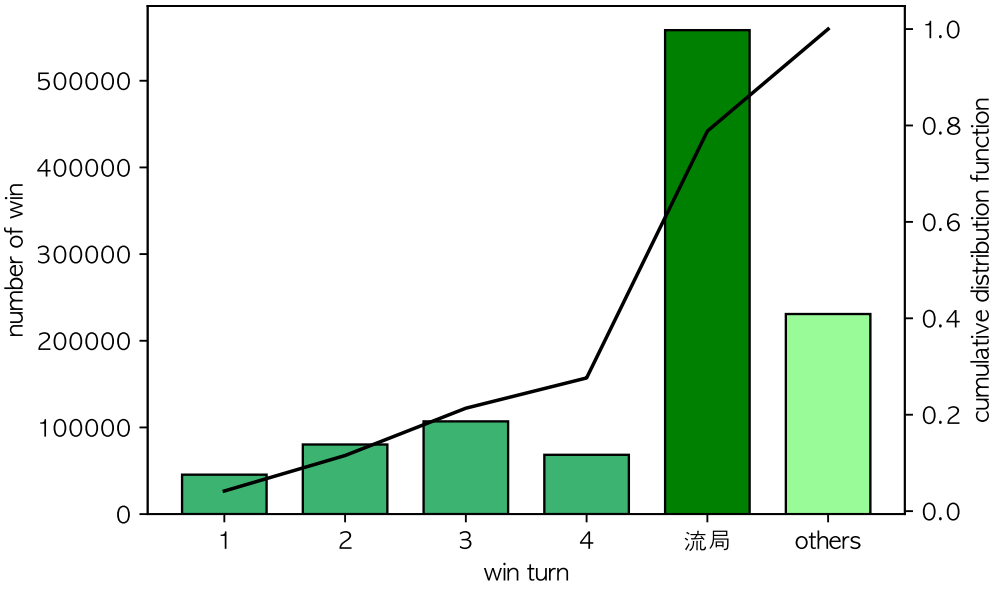


図 5.5　ランダムプレイヤとの五人プレイ対戦時の流局回数，他家のあがり回数（others），該当する順目でのあがり回数（ヒストグラム左軸）．累積分布関数（黒線右軸）．

のライブラリである OpenAI baselines[*8] のデフォルトである「Multi Layer Perceptron（MLP）」（隠れ層が 2 層で，ユニット数はそれぞれ 64）と，より表現力がある ResNet [20] をベースにした「resnet」の二つのモデルを使用した．

この resnet の具体的な層の構成を図 5.12 に示す．Conv1d の最初の引数はカーネルサイズを，二つ目の引数はフィルターの数を表す．最後の policy は行動の数，つまり各牌の種類数のユニットを持つ全結合層 1 つが入っている．その他のハイパーパラメータは機械学習ライブラリである tensorflow[*9] のデフォルト値に従った．このモデルはミニ麻雀の Conv1d モデルの結果を参考にして作成した．ゲームを対象にした先行研究の多くでは ResNet を用いる際は res_block の数は 10 個以上とすることが多く，例えば AlphaGo Zero [6] では 19 個あるいは 39 個が用いられている．しかし，本研究では一人すずめ雀プレイヤの方策を教師ありで層の数の異なる resnet に学習させる予備実験を行い，5 個以上では大差がなかったので，学習時間の短い 5 個を用いた．

手牌に関しては役ありミニ麻雀でも役なしミニ麻雀でも高い正解率になった手牌モード 2 を採用した．すなわち，入力には親，現在の局数，ドラ，自分の手牌，各プレイヤの累積得点，各プレイヤの捨て牌を使用しており，次のように表現する．

自分の手牌とドラに関しては，13 行 8 列，要素は 0 か 1 の二値の行列を用いる．行は牌の種類を表すが，9 索と發，發と中の間には 1 行入れるため 13 行となっている．その行にあたる牌が手

[*8] https://github.com/openai/baselines
[*9] https://www.tensorflow.org/

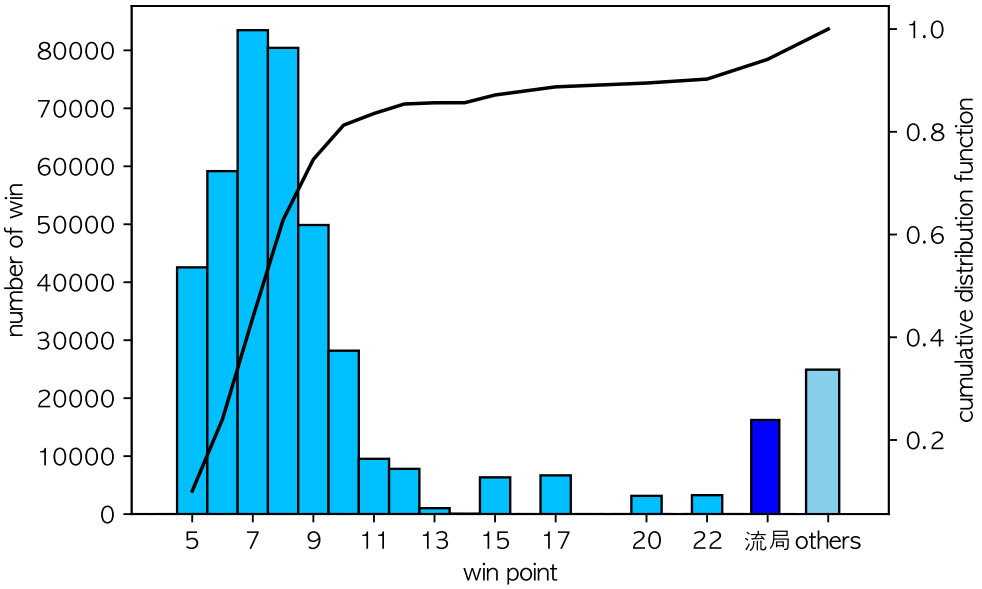


図 5.6　ランダムプレイヤとの二人プレイ対戦時の流局回数，他家のあがり回数（others），該当する得点でのあがり回数（ヒストグラム左軸）．累積分布関数（黒線右軸）．

牌に i 枚存在した時，1 列目から i 列目までを 1 とし，それ以外を 0 とする．ただし $i = 0$ の時は，1 から 4 列目まで全て 0 とする．5 列目は，その行にあたる牌で，かつ赤ドラとなっている牌を持っている時には 1，それ以外では 0 とする．6 列目に関しては，その行にあたる牌がドラ表示牌であったら 1，それ以外では 0 とする．7 行目に関しては，その行にあたる牌の赤牌がドラ表示牌であったら 1，それ以外では 0 とする．8 行目に関しては，10 行目と 12 行目は 0 とし，それ以外は 1 とする．ドラが 2 索で，自分の手牌が「23r344 發」であった時の入力の具体例を図 5.13 にあげる．

現在の局数，親に関しては one-hot エンコーディングとする．ゲームに参加する人数を p として，現在の局数は $4 \times p$ 次元，親は p 次元とし，当てはまる要素に 1 を，それ以外の要素には 0 をいれる．各プレイヤの累積得点に関して，0 から 100 点については 5 点ずつのバケットに分割して当てはまる要素を 1，それ以外を 0 とする．具体的には「0〜4 点」，「5〜9 点」，…，「95〜99 点」と分割する．0 点未満と 100 点以上はそれぞれ一つの次元でまとめるため，プレイヤ一人あたり合計で 22 次元となる．各プレイヤの捨て牌に関しては，その行にあたる牌が捨ててあれば 1，それ以外は 0 とする．ただし，親，現在の局数，各プレイヤの累積得点に関しては 13 行に渡って同じものを並べることで，全部で $13 \times (27 \times p + 8)$ のテンソルで入力を表せるようになる．

今回用いた方策の損失関数は式 (5.1) で定義されたものを用いる．

$$E_t \left[\max(-r_t(\theta)A_t, -\mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t) \right] \tag{5.1}$$

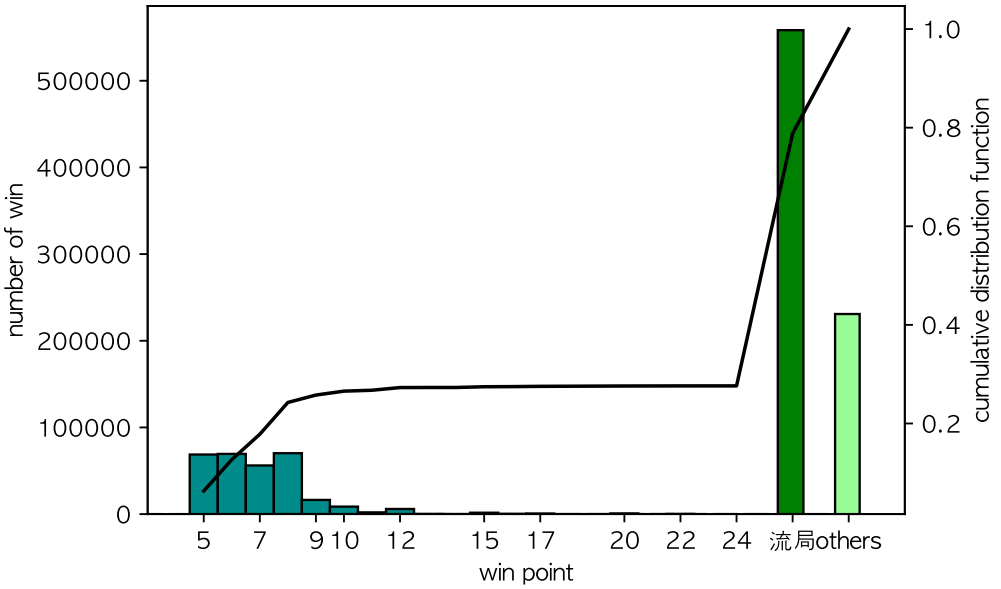


図 5.7　ランダムプレイヤとの五人プレイ対戦時の流局回数，他家のあがり回数（others），該当する得点でのあがり回数（ヒストグラム左軸）．累積分布関数（黒線右軸）．

$r_t(\theta)$ は，時刻 t における古いパラメータ θ' を使った方策と現在のパラメータ θ を使った方策の比（つまり $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)}$）を表し，$A_t$ は時刻 t における advantage を表し，$\mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)$ は $r_t(\theta)$ を $1-\epsilon$ から $1+\epsilon$ に制限することを表す．この clip には更新の前後で方策の値が大きく変わることを防ぐ目的があり，ϵ の値はデフォルト値である 0.2 を用いた．

プレイヤはあがれる時には必ずあがるものとし，自分以外のプレイヤは一人すずめ雀プレイヤと同じ行動を取るとした．プレイヤが行動をしてから，もう一度自分の手番が回ってくるまでを 1step とし，3×10^7 step 学習させた．学習時の resnet モデルでの policy loss と value loss について，二人プレイ時を図 5.14 に，五人プレイを図 5.15 に示す．

得られたモデルにおいて，強化学習で得られたプレイヤを 3×10^6step 対戦させた時の平均順位を表 5.2 に示す．この step 数は約 65000 ゲームに相当する．

表 5.2　MLP モデル，resnet モデルにおける平均順位．

設定＼モデル	MLP	resnet
二人プレイ	1.748	**1.528**
五人プレイ	3.501	**3.034**

二人プレイでは平均順位が 1.5 で同等，五人プレイでは平均順位が 3 で同等であると言える．こ

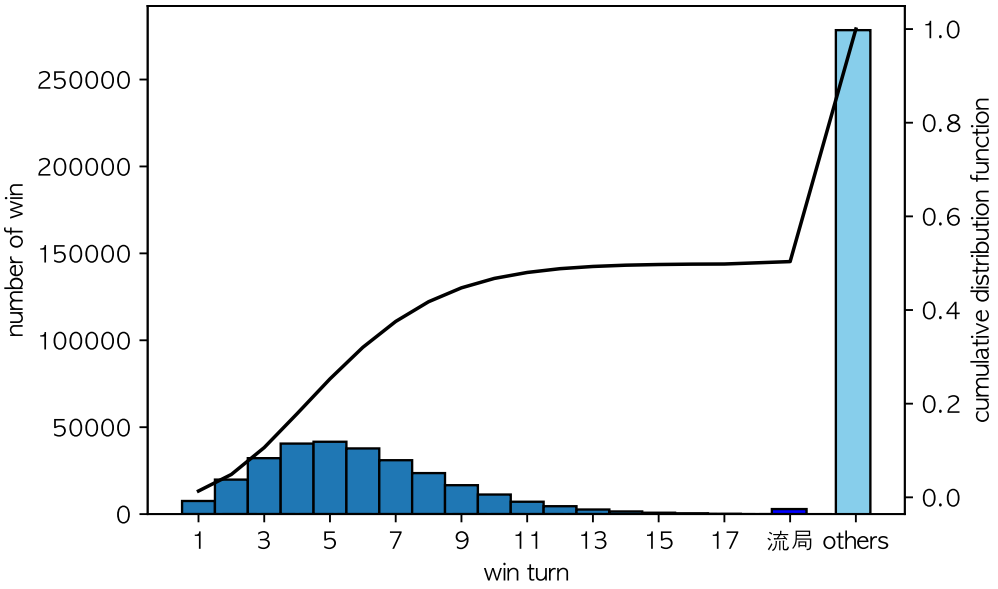


図 5.8　一人すずめ雀プレイヤとの二人プレイ対戦時の流局回数，他家のあがり回数（others），該当する順目でのあがり回数（ヒストグラム左軸）．累積分布関数（黒線右軸）．

の実験結果において，MLP を使用したモデルはどちらの実験設定においても一人すずめ雀プレイヤに対して負け越しているが，resnet を使用したモデルは一人すずめ雀プレイヤと同程度の実力を得たことがわかる．MLP はモデルの表現力が十分でないが，resnet はある程度の表現力を持つからであると考えられる．

5.4　Global Reward Predictor の作成

次に前節で得られたモデルを改良することを目標として，強化学習における良い報酬を出力するために最終結果を予測する Global Reward Predictor Φ を作る．この予測器の入力には，現在の局数 k，親，それまでの累積得点，その局の収支を含む特徴ベクトル x^k を用いる．なお Suphx ではそれらに加えて供託や連荘の回数という情報も含むが，すずめ雀においてはそれらは存在しないので含んでいない．具体的な表現方法としては以下である．

現在の局数や親というカテゴリカルな情報は，強化学習時の入力と同様に表す．その局までの累積得点やその局の得点はバケットに分割して当てはまる要素を 1，それ以外を 0 とする．累積得点については強化学習時と同様に表す．その局の得点については -21 点以下から 19 点以上までを 2 点刻みのバケットに分割するため，この情報は合計で 22 次元となる．

使用したモデルは図 5.16 のように，Suphx で使われたモデルだけでなく，RNN model と Dense model を追加した．GRU と RNN の引数はユニット数を，Dense の引数はユニット数と活性化関数を表す．RNN model に関しては GRU よりも単純な構造で置き換えた時にどうなるのかを調べ

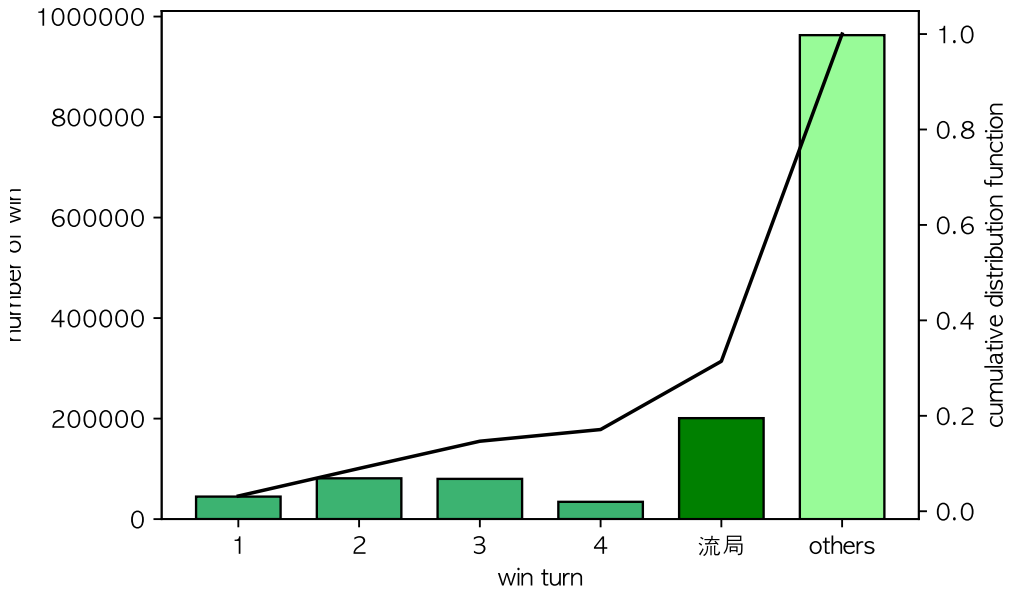


図 5.9　一人すずめ雀プレイヤとの五人プレイ対戦時の流局回数，他家のあがり回数（others），該当する順目でのあがり回数（ヒストグラム左軸）．累積分布関数（黒線右軸）．

るため，また Dense model に関しては，特徴ベクトル x^k には以前の局を踏まえた情報が含まれており，reccurent なモデルを使用しなくても良い可能性があると判断したためである．前述の一人すずめ雀プレイヤ同士の対戦のログ 3×10^6step 分（二人プレイでは 70079 ゲーム分，五人プレイでは 68058 ゲーム分）を学習に使用した．この学習では，損失関数は Suphx と同じ二乗誤差を使用する．

全データのうち 90 %を訓練データ，残りをバリデーションデータとした．学習時のハイパーパラメータのうち，エポック数は 100，バッチサイズは 128 とした．残りのハイパーパラメータは機械学習ライブラリである keras[*10] のデフォルト値に従った．

学習時の結果を図 5.17 と図 5.18 示す．図 5.17 では教師あり学習時の loss を，図 5.18 には validation loss を示す．

loss はどの手法においても下がっているが，一番下がったのは先行研究と同じモデルである GRU model であった．k 局目の情報である特徴ベクトル x^k には，それまでの累積得点，現在の親，現在の局数という情報が含まれているので，1 から $k-1$ 局までの情報を含んでいる．しかし，時系列データであることは変わりなく，RNN model や GRU model といった reccurent な情報を扱えるモデルの方が，loss が下がりやすいという結果であった．

そこで，Global Reward Predictor の各モデルを報酬に使用した強化学習によって，resnet のすずめ雀プレイヤを構築した．報酬以外の条件は先ほどと同じにする．得られたモデルを 3×10^6step

[*10] https://keras.io/ja/

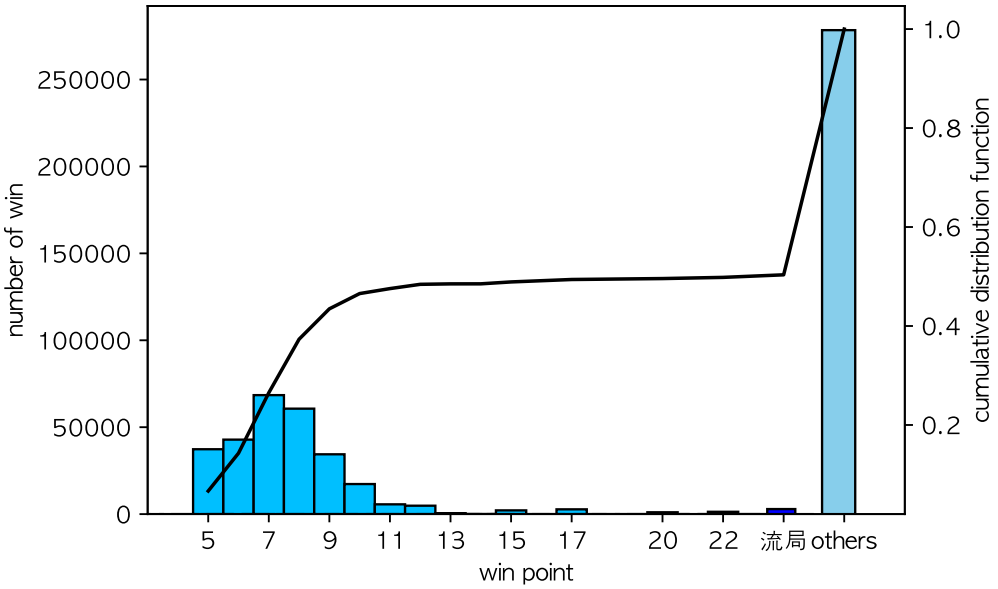


図 5.10　一人すずめ雀プレイヤとの二人プレイ対戦時の流局回数，他家のあがり回数（others），該当する得点でのあがり回数（ヒストグラム左軸）．累積分布関数（黒線右軸）．

だけ一人すずめ雀プレイヤと対戦させた．二人プレイ時の平均順位を表 5.3 に示す．なお表には比較のために MLP モデルを使用したすずめ雀プレイヤの平均順位も比較のために表示している．

表 5.3　二人プレイ時の各実験設定におけるすずめ雀プレイヤの平均順位．

報酬 / 設定	得点と順位点	Dense model	RNN model	GRU model
MLP	1.748	**1.708**	1.718	1.715
resnet	**1.528**	1.581	1.552	1.530

Global Reward Predictor が MLP という単純なモデルにおいてはすずめ雀でも効果的であったということ，また Suphx で提案されているモデル以外の簡単なモデルでも有効であることが分かった．しかし，ある程度の表現力をもつ resnet を使用した場合は，平均順位の改善には至らなかった．

この理由は複数考えられる．一つ目は Global Reward Predictor を使用しない場合，報酬に現在の順位を順位点に変換したものを使用しているが，これが強化学習においてある程度良い報酬になっていたという可能性である．この強化学習の目標は順位点の獲得であり，ある局における現在の順位は最終順位と大きく関係があるはずである．二つ目として，一人すずめ雀プレイヤが十分に強いという可能性である．一人すずめ雀プレイヤ同士を対戦させた時，自分が一度行動するのを 1 巡として，二人プレイ時では平均 5.8 巡，五人プレイ時では平均 2.43 巡で一局が終了する．すず

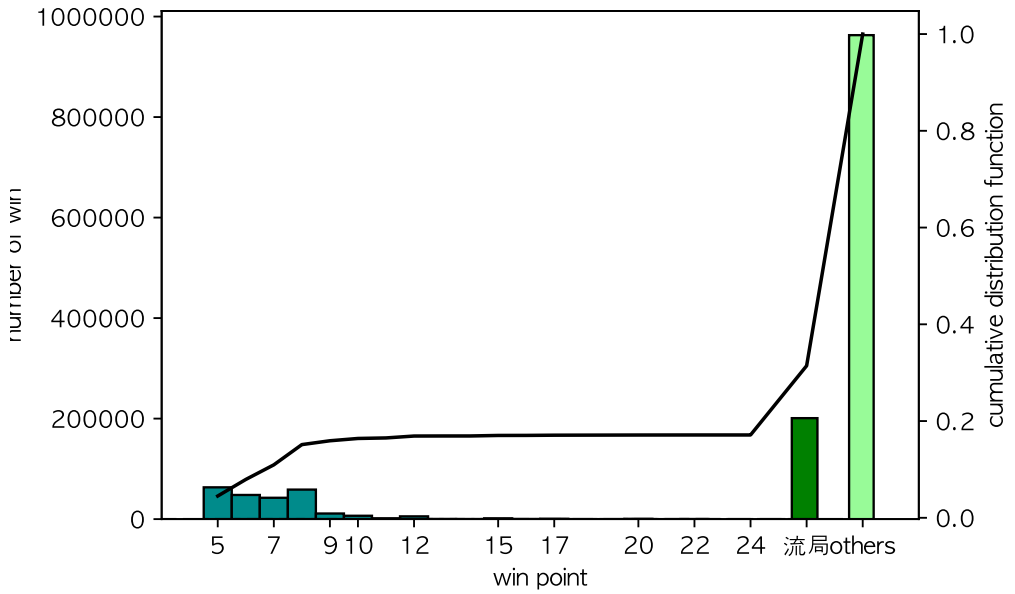


図 5.11　一人すずめ雀プレイヤとの五人プレイ対戦時の流局回数，他家のあがり回数（others），該当する得点でのあがり回数（ヒストグラム左軸）．累積分布関数（黒線右軸）．

め雀は通常の麻雀に比べて一局の間にできる行動が少ない．一人すずめ雀プレイヤは自分の手牌しか考慮に入れず，最善のあがりを目指すプレイヤであるが，それがすずめ雀にとっては最適に近い戦略であるという可能性が考えられる．

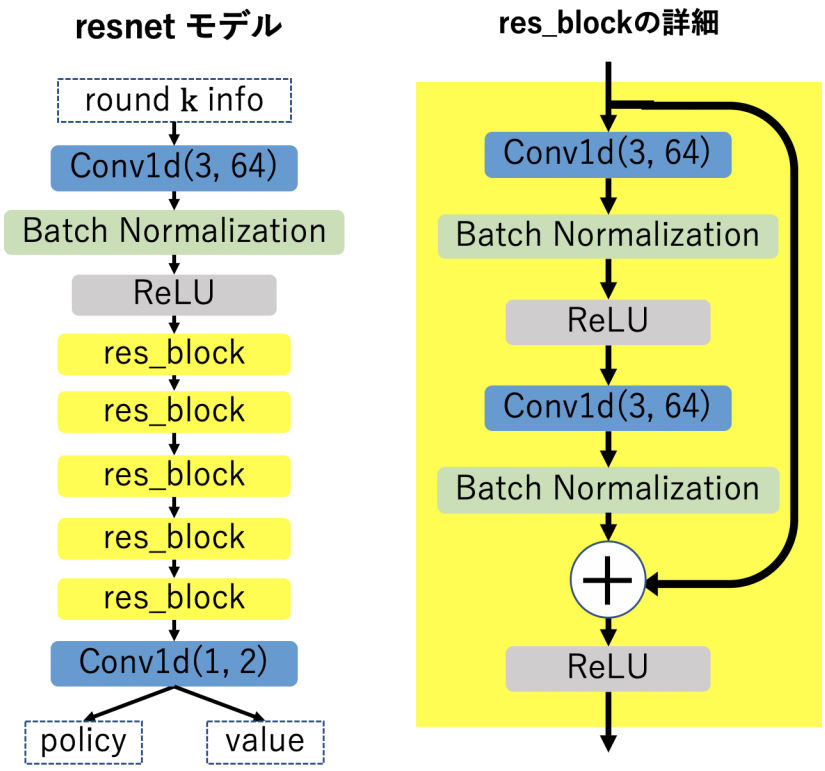


図 5.12　resnet モデルの各層の構成.

		1枚以上	2枚以上	3枚以上	4枚	赤を含むか	ドラか	赤牌がドラか	牌かどうか
牌の種類	1	0	0	0	0	0	0	0	**1**
	2	**1**	0	0	0	0	**1**	0	**1**
	3	**1**	**1**	0	0	**1**	0	0	**1**
	4	**1**	**1**	0	0	0	0	0	**1**
	5	0	0	0	0	0	0	0	**1**
	⋮								
	9	0	0	0	0	0	0	0	**1**
	blank	0	0	0	0	0	0	0	0
	發	**1**	0	0	0	0	0	0	**1**
	blank	0	0	0	0	0	0	0	0
	中	0	0	0	0	0	0	0	**1**

図 5.13　ドラが 2 で自分の手牌が「23r344 發」であった時の入力例.

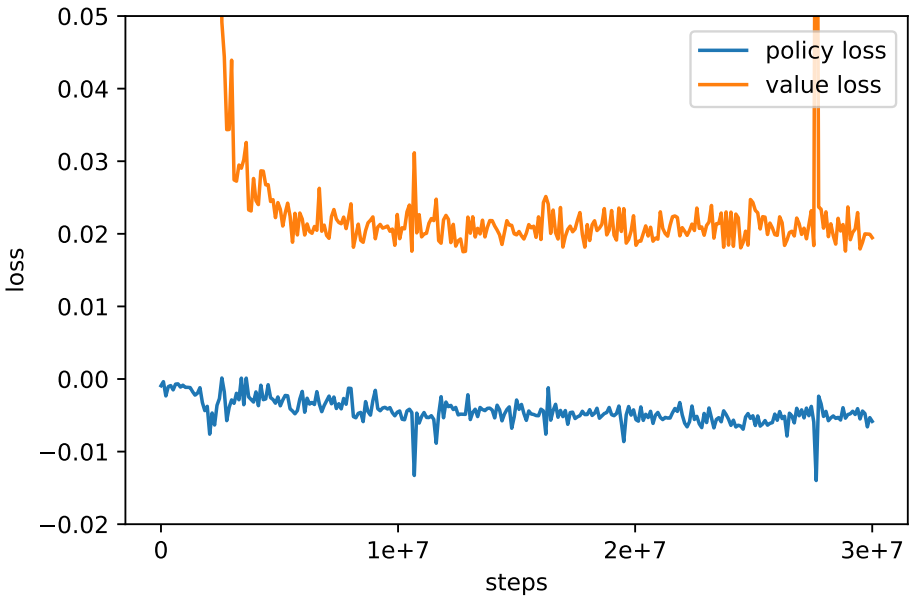


図 5.14　二人プレイ時の resnet モデルでの policy loss と value loss.

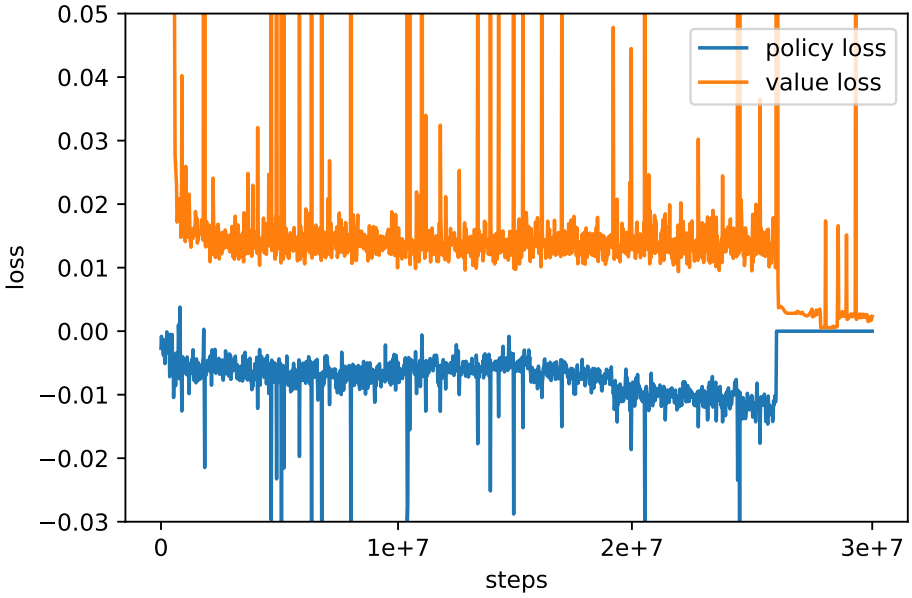


図 5.15　五人プレイ時の resnet モデルでの policy loss と value loss.

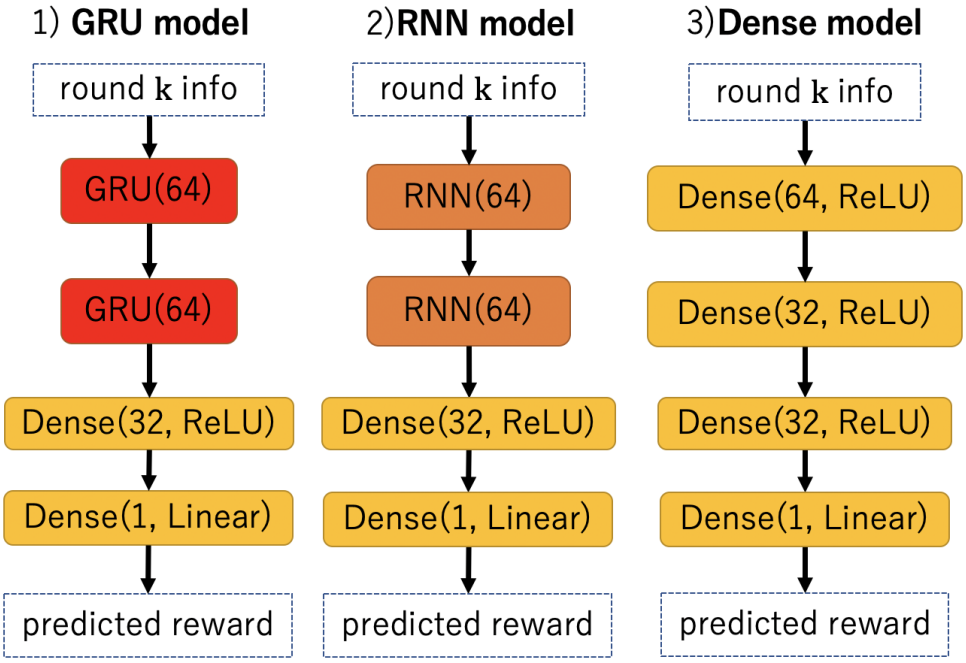


図 5.16　Global Reward Predictor で使用したモデルの概要.

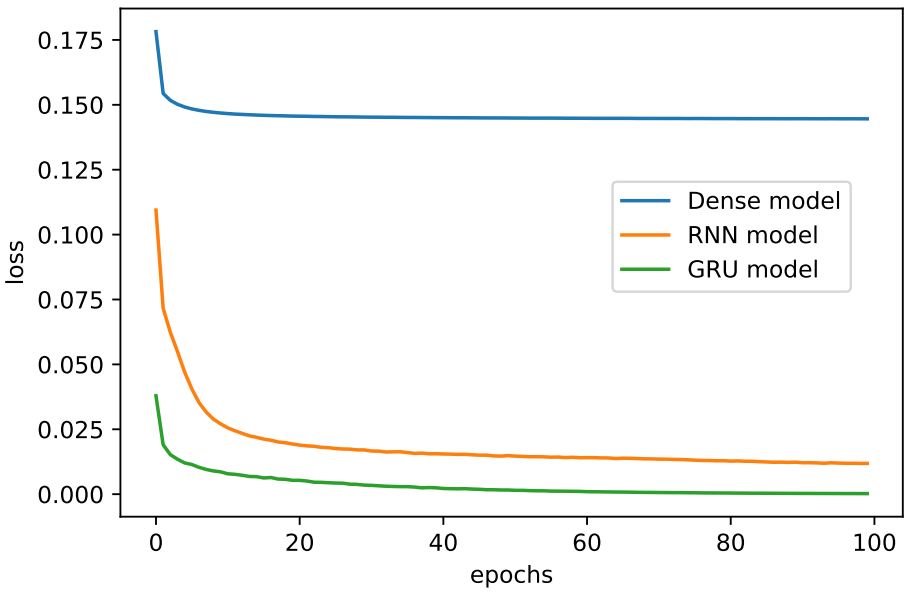


図 5.17　Global Reward Predictor の学習時の loss.

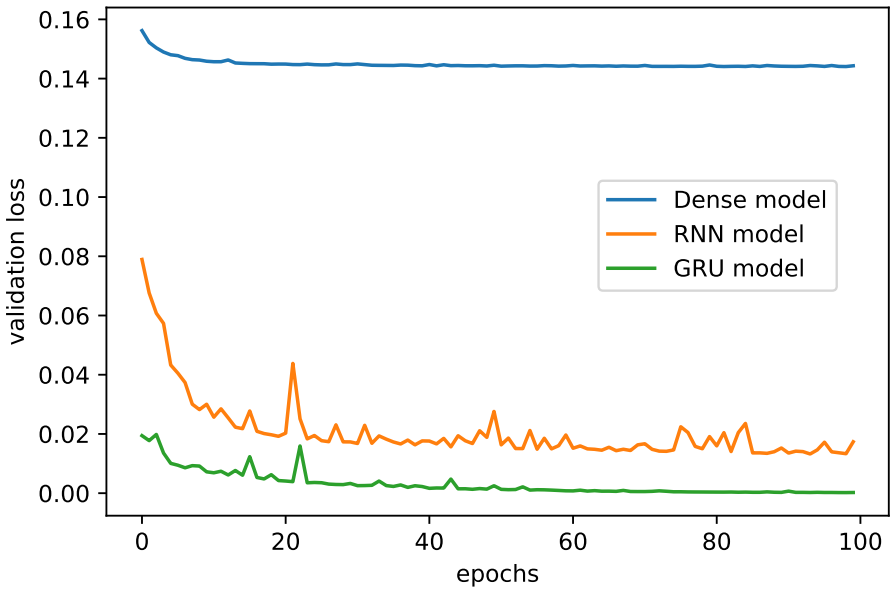


図 5.18　Global Reward Predictor の学習時の val_loss.

第 6 章

まとめと今後の課題

本研究では，手生成の特徴量（hand-crafted features）や上級者の牌譜といったゲームに関する人間の事前知識を使わずに，麻雀において自己対戦による強化学習のみで人間の上級者を超える実力を得ることを本研究の目標として実験を行なった．

ミニ麻雀の実験では，理論的な最善手が求められる小さいゲームを考えて，その割引収益の期待値を求めた．その期待値を元にした教師あり学習を行い，ニューラルネットワークモデルと手牌の表現方法の評価を行なった．ニューラルネットワークモデルの構成には Optuna という自動最適化ツールを用いた．提案したモデルにおいては先行研究よりも 2 から 4 ポイント高い正解率を得ることができた．また手牌の表現方法に関しては手牌モード 2 が役ありでも役無しでも高い正解率を得た．

今回はミニ麻雀の中でも簡単な八枚麻雀のみで実験を行ったが，牌の種類や手牌の枚数を増やしたり，役をさらに増やしたりすることが今後の課題となる．その時に状態数が増える結果として理論的な最善手を求めるのが難しい場合には，探索を用いて近似値を求め，その値を教師データとすることも必要になると考えられる．

またすずめ雀においては，手生成の特徴量や上級者の牌譜といったゲームに関する人間の事前知識を使わずに深層強化学習を用いたすずめ雀プレイヤを構築した．このプレイヤは，割引収益の期待値の最も高い牌を切ることのできる「一人すずめ雀プレイヤ」と同程度の実力を得ることができた．また強化学習に用いるモデルの表現力が十分でなく，まだ改善の余地があるようなすずめ雀プレイヤにおいては，Global Reward Predictor が効果的であることがわかったが，強化学習のモデルに resnet を使用し，すでにある程度の強さをもつプレイヤにとっては Global Reward Predictor が効果的な手法とは言えない可能性があることがわかった．

麻雀では，様々な要因によって学習が困難になる．例えば「プレイヤから見えない牌が多く，報酬関数を作るのが難しい」や「鳴きで順番が変わるのでゲーム木のサイズが膨大となり，MCTS などの従来手法が使えない」などである．上記の問題を解決する手法として，Suphx で提案されている他の学習手法やそれを改善した手法があり，今後はそれをすずめ雀で評価していきたい．すずめ雀で十分な効果が得られた手法は，実際の麻雀にも適用できる可能性がある．また本研究の内容は乱数の初期シードの影響をうける．今回は一つのシードのみでしか実験を行っておらず，複数シー

ドで実験をするということも今後の課題としたい．また，すずめ雀を通常の麻雀と比べると「ポン，チー，カンなどの鳴きを考慮した打牌選択」という要素がない．この通常の麻雀との大きな差異であるためここを埋めるような手法も考える必要がある．また割引率 γ も本研究においては 0.9 としているが，五人プレイ時では最大でも 4 回しかツモれないということもあり，より低い値に設定する方が適当である可能性が高い．より適当な値を考慮に入れて実験することも今後の課題となる．

なお，本研究で用いたプログラムは GitHub(ミニ麻雀は `https://github.com/minnsou/gpw2019`，すずめ雀は `https://github.com/minnsou/suzume-jong`) で公開している．

謝辞

本研究に際して様々なご指導を頂きました田中哲朗准教授に心より御礼申し上げます．先生は一緒に環境構築並びにコーディングを手伝っていただいたこともあり，本当に感謝しております．また，ミーティング等で助言をいただいた田中研究室の皆様，私の発表を聞いて適切なコメントをくださった同じゼミの金子研究室の皆様にも感謝いたします．

参考文献

[1] Noam Brown and Tuomas Sandholm. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, Vol. 359, No. 6374, pp. 418–424, 2018.

[2] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, Vol. 575, No. 7782, pp. 350–354, 2019.

[3] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019. [Online]. Available: `https://arxiv.org/abs/1912.06680`.

[4] Junjie Li, Sotetsu Koyamada, Qiwei Ye, Guoqing Liu, Chao Wang, Ruihan Yang, Li Zhao, Tao Qin, Tie-Yan Liu, and Hsiao-Wuen Hon. Suphx: Mastering mahjong with deep reinforcement learning. *arXiv preprint arXiv:2003.13590*, 2020. [Online]. Available: `https://arxiv.org/abs/2003.13590`.

[5] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, Vol. 529, No. 7587, pp. 484–489, 2016.

[6] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, Vol. 550, No. 7676, pp. 354–359, 2017.

[7] 築地毅, 柴原一友. CNN 麻雀 ー麻雀向け CNN 構成の有効性ー. ゲームプログラミングワークショップ 2017 論文集, pp. 163–170, nov 2017.

[8] Shiqi Gao, Fuminori Okuya, Yoshihiro Kawahara, and Yoshimasa Tsuruoka. Supervised learning of imperfect information data in the game of mahjong via deep convolutional neural networks. *Information Processing Society of Japan*, 2018.

[9] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of*

the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019.

[10] 清水大志, 田中哲朗. 麻雀のポリシー関数に適したネットワークモデルの構築と評価. ゲームプログラミングワークショップ 2019 論文集, pp. 165–171, nov 2019.

[11] K.Maruta and T.Shinozaki. Suzume-jong, 2018. [Online]. Available: `https://sugorokuya.jp/p/suzume-jong/`.

[12] 清水大志, 田中哲朗. 深層強化学習を用いた麻雀プレイヤの構築. ゲームプログラミングワークショップ 2020 論文集, pp. 147–154, nov 2020.

[13] 水上直紀, 中張遼太郎, 浦晃, 三輪誠, 鶴岡慶雅, 近山隆. 多人数性を分割した教師付き学習による 4 人麻雀プログラムの実現. 情報処理学会論文誌, Vol. 55, No. 11, pp. 2410–2420, 2014.

[14] 水上直紀, 鶴岡慶雅. 期待最終順位に基づくコンピュータ麻雀プレイヤの構築. ゲームプログラミングワークショップ 2015 論文集, pp. 179–186, oct 2015.

[15] 水上直紀, 鶴岡慶雅. 強化学習を用いた効率的な和了を行う麻雀プレイヤ. ゲームプログラミングワークショップ 2016 論文集, pp. 81–88, oct 2016.

[16] 水上直紀, 鶴岡慶雅. 自動対戦棋譜の教師あり学習による翻数予測に基づく麻雀プレイヤ. 情報処理学会論文誌, Vol. 60, No. 7, pp. 1325–1336, jul 2019.

[17] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[18] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, Vol. 24, pp. 2546–2554, 2011.

[19] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. [Online]. Available: `https://arxiv.org/abs/1707.06347v2`.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.